Commutative Data Automata

Zhilin Wu

State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
wuzl@ios.ac.cn

Abstract

Formalisms over infinite alphabets have recently received much focus in the community of theoretical computer science. Data automata is a formal model for words over infinite alphabets proposed by Bojanczyk, Muscholl, Schwentick et. al. in 2006. A data automaton consists of two parts, a nondeterministic letter-to-letter transducer, and a class condition specified by a finite automaton over the output alphabet of the transducer, which acts as a condition on the subsequence of the outputs of the transducer in every class, namely, in every maximal set of positions with the same data value. It is open whether the nonemptiness of data automata can be decided with elementary complexity. Very recently, a restriction of data automata with elementary complexity, called weak data automata, was proposed by Kara, Schwentick and Tan and its nonemptiness problem was shown to be in 2-NEXPTIME. In weak data automata, the class conditions are specified by some simple constraints on the number of occurrences of labels occurring in every class. The aim of this paper is to demonstrate that the commutativity of class conditions is the genuine reason accounting for the elementary complexity of weak data automata. For this purpose, we define and investigate commutative data automata, which are data automata with class conditions restricted to commutative regular languages. We show that while the expressive power of commutative data automata is strictly stronger than that of weak data automata, the nonemptiness problem of this model can still be decided with elementary complexity, more precisely, in 3-NEXPTIME. In addition, we extend the results to data ω-words and prove that the nonemptiness of commutative Büchi data automata can be decided in 4-NEXPTIME. We also provide logical characterizations for commutative (Büchi) data automata, similar to those for weak (Büchi) data automata.

1998 ACM Subject Classification F.1.1, F4

Keywords and phrases Data Automata, Commutative regular languages, Presburger arithmetic, Existential second-order logic, Büchi automata

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

With the momentums from the XML document processing and the verification of computer programs, formalisms over infinite alphabets have been intensively investigated in recent years. In the database community, XML documents are usually represented by trees, where the nodes can have tags together with several attributes e.g. identifiers. While the tags are from a finite set, the attributes may take values from some infinite domains. On the other hand, in the verification community, take concurrent systems as an example, if there are unbounded number of processes in the system, then the behavior of the global system consists of the sequences of observed events attached with the process identifiers.

With these motivations, researchers in the two communities have investigated various formalisms over infinite alphabets, to name a few, register automata ([10]), pebble automata ([14]), data automata ([3]), XPath with data values ([7, 8]), LTL with freeze quantifiers ([6]), as well as two-variable logic interpreted on the words or trees over infinite alphabets ([3, 2]). A survey on this topic can be found in [16].

By infinite alphabet, we mean $\Sigma \times \mathbb{D}$, with Σ a finite set of tags (labels) and $\mathbb{D}$ an infinite data domain. Words and trees over the alphabet $\Sigma \times \mathbb{D}$ are called data words and data trees.

Data automata was introduced by Bojanczyk, Muscholl, Schwentick, et. al. in [3] to prove the decidability of two-variable logic over data words. A data automaton $\mathscr{D}$ over data words consists of two parts, a nondeterministic letter-to-letter transducer $\mathscr{A}$, and a class condition specified by a finite automaton $\mathscr{B}$ over the output alphabet of $\mathscr{A}$, which acts as a condition on the subsequence of the outputs of $\mathscr{A}$ in every class, namely, every maximal set of positions with the same data value. By a reduction to the reachability problem of Petri nets (also called multicounter machines), the nonemptiness of data automata was shown to be decidable. On the other hand, data automata is also powerful enough to simulate petri nets easily. Since it is a well-known open problem whether the complexity of the reachability problem for petri nets is elementary, it is also not known whether the nonemptiness of data automata can be decided with elementary complexity.

Aiming at lowering down the complexity of data automata, a restriction of data automata, called weak data automata, was introduced and investigated very recently by Kara, Schwentick, and Tan ([11]). In weak data automata, the class conditions are replaced by some simple constraints on the number of occurrences of labels occurring in every class. The nonemptiness of weak data automata can be decided with elementary complexity, more precisely, can be decided in 2-NEXPTIME.

By comparing data automata with weak data automata, we notice that to simulate Petri nets in data automata, the ability to express the property $L_{a<b}$, "for every occurrence of a, there is an occurrence of b on the right with the same data value", is crucial; on the other hand, as shown in [11], $L_{a<b}$ is not expressible in weak data automata. It is a simple observation that $L_{a<b}$ is a non-commutative language while the class conditions of weak data automata are commutative. This suggests that the commutativity of class conditions might be the *genuine* reason accounting for the elementary complexity of weak data automata. With this observation, we are motivated to define and investigate commutative data automata, which are data automata with class conditions restricted to commutative regular languages. We would like to see that the nonemptiness of commutative data automata can still be decided with elementary complexity, even though they have stronger class conditions than weak data automata. This is indeed the case, as we will show in this paper.

More specifically, the contributions of this paper consist of the following three aspects.

1. At first, we investigate the expressibility issues of commutative data automata. We show that the expressive power of commutative data automata lies strictly between that of data automata and weak data automata. In addition, commutative data automata are closed under intersection and union, but not closed under complementation. We also present a logical characterization of commutative data automata, similar to that for weak data automata.
2. The nonemptiness of commutative data automata can be decided in 3-NEXPTIME, which is the main result of this paper.
3. At last, we extend the results to the data ω-words. We define commutative Büchi data automata and prove that the nonemptiness of commutative Büchi data automata can be decided in 4-NEXPTIME.

The main ideas of most of the proofs in this paper come from those for weak data automata ([11, 5]). Nevertheless, some proof steps become much more complicated as a result of the additional intricacies brought upon by the stronger class conditions in commutative data automata.

Related work.

Several variants of data automata have been investigated. Bojanczyk and Lasota proposed an (undecidable) extension of data automata, called class automata, to capture the full XPath with data values; in addition, they established the correspondences of various class conditions of class automata with the various models of counter machines ([1]). Wu continued this line of research by introducing another decidable extension of data automata and establishing the correspondence with priority multicounter machines ([19]). There is another automata model, called class counting automata, relevant to this paper. Class counting automata was proposed by Manuel and Ramanujan in [12]. In class counting automata, each data value is assigned a counter; and in each transition step, if the value of the counter corresponding to the current data value satisfies some constraint, then the value of the counter is updated according to a prescribed instruction. The nonemptiness of class counting automata was shown to be EXPSPACE-complete. Nevertheless, the expressive power of class counting automata is relatively weak, for instance, the property "Each data value occurs exactly twice" cannot be expressed by class counting automata, while this property can be easily expressed by commutative data automata.

Commutative regular languages have been investigated by many researchers. Pin presented a counting characterization of the expressibility of commutative regular languages ([15]). Gomez and Alvarez investigated how commutative regular languages can be learned from positive and negative examples ([9]). Chrobak, Martinez and To proposed polynomial time algorithms to obtain regular expressions from nondeterministic finite automata over unary alphabets ([4, 13, 18]).

The rest of this paper is organized as follows. Some preliminaries are given in the next section. Then in Section 3, the expressibility of commutative data automata are investigated. Section 4 includes the main result of this paper, a 3-NEXPTIME algorithm for the nonemptiness of commutative data automata. The results are extended to data ω-words in Section 5. Finally in Section 6, some conclusions are given and the future work are discussed. The missing proofs can be found in the appendix.

2 Preliminaries

In this section, we fix the notations used in this paper.

Let Σ be a finite alphabet. A finite word over Σ is an element of Σ^* and a ω-word over Σ is an element of Σ^ω.

2.1 Presburger formulas and Commutative regular languages

Existential Presburger formulas (EP formulas) over a variable set X are formulas of the form $\exists \bar{x} \varphi$, where φ is a quantifier-free Presburger formula, i.e. a Boolean combination of atomic formulas of the form $t \geq c$, or $t \leq c$, or $t = c$, or $t \equiv r \bmod p$, where $c, r, p \in \mathbb{N}$, $p \geq 2, 0 \leq r < p$ and t is a term defined by $t := c (c \in \mathbb{N}) \mid cx (c \in \mathbb{N}, x \in X) \mid t_1 + t_2 \mid t_1 - t_2$.

Suppose $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ and $v \in \Sigma^*$. The *Parikh image* of v, denoted by $\mathsf{Parikh}(v)$, is a k-tuple $(\#_v(\sigma_1), \ldots, \#_v(\sigma_k))$, where for each $i : 1 \leq i \leq k$, $\#_v(\sigma_i)$ is the number of occurrences of σ_i in v. Let $V_\Sigma = \{x_{\sigma_1}, \ldots, x_{\sigma_k}\}$ and φ be a EP formula with free variables from V_Σ. The word v is said to satisfy φ, denoted by $v \models \varphi$, iff $\varphi[\mathsf{Parikh}(v)]$ holds. The *language defined by* φ, denoted by $\mathcal{L}(\varphi)$, is the set of words $v \in \Sigma^*$ such that $v \models \varphi$.

A *Presburger automaton* over the alphabet Σ is a binary tuple $(\mathscr{A}, \varphi)$, where $\mathscr{A}$ is a finite automaton over the alphabet Σ and φ is a EP formula with free variables from V_Σ. A word $v \in \Sigma^*$ is accepted by a Presburger automaton $(\mathscr{A}, \varphi)$ iff v is accepted by $\mathscr{A}$ and at the same time $v \models \varphi[\mathsf{Parikh}(v)]$. The language accepted by a Presburger automaton $(\mathscr{A}, \varphi)$, denoted by $\mathcal{L}((\mathscr{A}, \varphi))$, is the set of words accepted by $(\mathscr{A}, \varphi)$.

▶ **Theorem 1** ([17]). *The nonemptiness of Presburger automata can be decided in NP.*

Let L be a language over the alphabet Σ. Then L is *commutative* iff for any $u, v \in \Sigma^*$, $uv \in L$ iff $vu \in L$. Commutative regular languages have a characterization in quantifier-free simple Presburger formulas defined in the following.

Quantifier-free simple Presburger formulas (QFSP formulas) over a variable set X are Boolean combinations of atomic formulas of the form $x_1 + \cdots + x_n \le c$, or $x_1 + \cdots + x_n \ge c$, or $x_1 + \cdots + x_n = c$, or $x_1 + \cdots + x_n \equiv r \bmod p$, where $x_1, \ldots, x_n \in X$, $c, r, p \in \mathbb{N}$, and $0 \le r < p$.

Let $V_\Sigma = \{x_{\sigma_1}, \ldots, x_{\sigma_k}\}$ and φ be a QFSP formula over the variable set V_Σ. Similar to EP formulas, we can define $\mathcal{L}(\varphi)$, the language defined by φ.

For a set of variables $\{x_1, \ldots, x_k\}$, we use the notation $\varphi(x_1, \ldots, x_k)$ to denote a EP or QFSP formula φ with the free variables from $\{x_1, \ldots, x_k\}$.

▶ Proposition 2 ([15]). *Let L be a regular language over the alphabet $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$. Then L is commutative iff L is defined by a QFSP formula $\varphi(x_{\sigma_1}, \ldots, x_{\sigma_k})$.*

The *size* of a EP or QFSP formula φ, denoted by $|\varphi|$, is defined as the length of a binary encoding of φ (where the constants c, r and p are encoded in binary).

▶ Proposition 3. *Let $\varphi(x_1, \ldots, x_k)$ be a QFSP formula. Then there exists an exponential-time algorithm to transform φ into a QFSP formula $\bigvee\limits_{i:1 \le i \le m} \varphi_i$ of size $2^{O(k|\varphi|)}$ such that there is $p_0 : 2 \le p_0 \le 2^{|\varphi|}$ satisfying that*

- *each φ_i is of the form $\bigwedge\limits_{1 \le j \le k} \varphi_{i,j}$;*
- *for each $j : 1 \le j \le k$, $\varphi_{i,j}$ is equal to $x_j = c_{i,j}$ or $x_j \ge p_0 \wedge x_j \equiv r_{i,j} \bmod p_0$ for $c_{i,j}, r_{ij} : 0 \le c_{i,j}, r_{i,j} < p_0$;*
- *in addition, those φ_i's are mutually exclusive.*

For a QFSP formula $\varphi(x_1, \ldots, x_k)$, the number p_0 and the QFSP formula $\bigvee\limits_{i:1 \le i \le m} \varphi_i$ in Proposition 3 are called respectively the *normalization* number and the *normal form* of φ.

2.2 Data words, two-variable logic and data automata

Let Σ be a finite alphabet and $\mathbb{D}$ be an infinite set of data values. A *data word* over Σ is an element of $(\Sigma \times \mathbb{D})^*$ and a *data ω-word* is an element of $(\Sigma \times \mathbb{D})^\omega$. Let $\sigma \in \Sigma$, a position in a data word or a data ω-word is called a *σ-position* if the position is labelled by σ.

Given a data (finite or ω) word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\ldots$, the *projection* of w to the finite alphabet Σ, denoted by $\mathsf{Proj}(w)$, is the (finite or ω) word $\sigma_1\sigma_2\ldots$. Let X be a set of positions in a word w, we use $w|_X$ to denote the restriction of w to the positions in X. Similarly, $w|_X$ can be defined for ω-words, data words and data ω-words.

Let $FO(+1, \sim, \Sigma)$ denote the first-order logic with the following atomic formulas, $\sigma(x)$ (where $\sigma \in \Sigma$), $x = y$, $x + 1 = y$, and $x \sim y$. Two positions x, y satisfy $x + 1 = y$ if y is the successor of the position x, and two positions satisfy $x \sim y$ if they have the same data value. Let $FO^2(+1, \sim, \Sigma)$ denote the two-variable fragment of $FO(+1, \sim, \Sigma)$. In addition, let $EMSO^2(+1, \sim, \Sigma)$ denote the extension of $FO^2(+1, \sim, \Sigma)$ by existential monadic second-order quantifiers in front of the $FO^2(+1, \sim, \Sigma)$ formulas.

A *class* of a data (finite or ω) word w is a maximal set of positions in w with the same data value. Given a class X of a data word w, the *class string* of w corresponding to X is $\mathsf{Proj}(w|_X)$, the projection of $w|_X$.

Let $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\dots\binom{\sigma_n}{d_n}$ be a data word and φ be a QFSP formula over the variable set V_Σ. Then w is said to *satisfy the class condition* φ, denoted by $w \models_c \varphi$, if for each class X of w, $\mathsf{Proj}(w|_X) \models \varphi[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$.

Given a data word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\dots\binom{\sigma_n}{d_n}$, the *profile word* of w, denoted by $\mathsf{Profile}(w)$, is a word $(\sigma_1, s_1)\dots(\sigma_n, s_n)$ over the alphabet $\Sigma \times \{\bot, \top\}$ such that for each $i : 1 \leq i < n$, $s_i = \top$ (resp. $s_i = \bot$) iff $d_i = d_{i+1}$ (resp. $d_i \neq d_{i+1}$), and $s_n = \bot$ by convention.

A *data automaton* (DA) $\mathscr{D}$ is a tuple $(\mathscr{A}, \mathscr{B})$, where $\mathscr{A} = (Q_1, \Sigma \times \{\bot, \top\}, \Gamma, \delta_1, q_{0,1}, F_1)$ is a nondeterministic letter-to-letter transducer with the input alphabet $\Sigma \times \{\bot, \top\}$ and the output alphabet Γ, and $\mathscr{B} = (Q_2, \Gamma, \delta_2, q_{0,2}, F_2)$ is a finite automaton over the alphabet Γ.

A data word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\dots\binom{\sigma_n}{d_n}$ is *accepted* by a data automaton $\mathscr{D} = (\mathscr{A}, \mathscr{B})$ iff there is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(w)$ which produces a word $\gamma_1 \dots \gamma_n$ such that for each class X of $w' = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2}\dots\binom{\gamma_n}{d_n}$, the class string $\mathsf{Proj}(w'|_X)$ is accepted by $\mathscr{B}$.

The *data language* defined by a data automaton $\mathscr{D}$, denoted by $\mathcal{L}(\mathscr{D})$, is the set of data words accepted by $\mathscr{D}$.

A *weak data automaton* (WDA) is a data automaton $(\mathscr{A}, \mathscr{C})$, with the class condition $\mathscr{C}$ specified by a collection of

- key constraints of the form $\mathsf{Key}(\gamma)$ (where $\gamma \in \Gamma$), interpreted as "every two γ-positions have different data values",
- inclusion constraints of the form $D(\gamma) \subseteq \bigcup_{\gamma' \in R} D(\gamma')$ (where $\gamma \in \Gamma, R \subseteq \Gamma$), interpreted as "for every data value occurring in a γ-position, there is $\gamma' \in R$ such that the data value also occurs in a γ'-position",
- and denial constraints of the form $D(\gamma) \cap D(\gamma') = \emptyset$ (where $\gamma, \gamma' \in \Gamma$), interpreted as "no data value occurs in both a γ-position and a γ'-position".

A data word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\dots\binom{\sigma_n}{d_n}$ is accepted by a weak data automaton $\mathscr{D} = (\mathscr{A}, \mathscr{C})$ iff there is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(w)$ which produces a word $\gamma_1 \dots \gamma_n$ such that the data word $w' = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2}\dots\binom{\gamma_n}{d_n}$ satisfies all the constraints in $\mathscr{C}$.

A *commutative data automaton* (CDA) $\mathscr{D}$ is a binary tuple $(\mathscr{A}, \varphi)$ such that $\mathscr{A} = (Q, \Sigma \times \{\bot, \top\}, \Gamma, \delta, q_0, F)$ is a letter-to-letter transducer and φ is a QFSP formula over the variable set V_Γ.

A data word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\dots\binom{\sigma_n}{d_n}$ is accepted by a commutative data automaton $\mathscr{D} = (\mathscr{A}, \varphi)$ iff there is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(w)$ which produces a word $\gamma_1 \dots \gamma_n$ such that the data word $w' = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2}\dots\binom{\gamma_n}{d_n}$ satisfies that $w' \models_c \varphi$.

▶ Remark. We choose to define the class conditions of commutative data automata by QFSP formulas, instead of finite automata with commutative transition relations. The main purpose of this choice is to ease the extension of the results to data ω-words (c.f. Section 5). ◀

From any WDA $(\mathscr{A}, \mathscr{C})$, an equivalent CDA $(\mathscr{A}, \varphi_{\mathscr{C}})$ can be constructed such that $\varphi_{\mathscr{C}} := \bigwedge\limits_{C \in \mathscr{C}} \varphi_C$, where φ_C is defined as follows,

- if C is of the form $\mathsf{Key}(\gamma)$, then $\varphi_C := x_\gamma \leq 1$,
- if C is of the form $D(\gamma) \subseteq \bigcup\limits_{\gamma' \in R} D(\gamma')$, then $\varphi_C := x_\gamma \geq 1 \rightarrow \sum\limits_{\gamma' \in R} x_{\gamma'} \geq 1$,
- if C is of the form $D(\gamma) \cap D(\gamma') = \emptyset$, then $\varphi_C := x_\gamma \geq 1 \rightarrow x_{\gamma'} = 0$.

▶ Remark. According to the above reduction of WDA to CDA, we remark that in some sense, CDA = WDA + Modular constraints in class conditions.

3 Expressiveness

In this section, we first show that the expressibility of CDA lies strictly between WDA and DA, then we discuss the closure properties of CDA and provide a logical characterization of CDA.

▶ **Theorem 4.** *WDA < CDA < DA.*

Proof.
CDA < DA.

It was shown in [11] that the language "for every occurrence of a, there is an occurrence of b on the right with the same data value" cannot be expressed in WDA. The same proof can be applied to show that the language is not expressible in CDA. On the other hand, it is easy to see that the language can be defined by a DA.
WDA < CDA.

It is easy to see that the language "In each class of the data word, the letter a occurs an even number of times" is expressible in CDA. By some pumping argument, we can show that the language is not expressible in WDA. ◀

Similar to data automata, commutative data automata are not closed under complementation.

▶ **Theorem 5.** *CDAs are closed under union and intersection, but not closed under complementation.*

In the following, we define $EMSO^2_{\#}(+1, \sim, \Sigma)$, a counting extension of $EMSO^2(+1, \sim, \Sigma)$, and show that it is expressively equivalent to CDA.

The logic $EMSO^2_{\#}(+1, \sim, \Sigma)$ includes all the formulas of the form $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$, such that $\varphi \in FO^2(+1, \sim, \Sigma, R_1, \dots, R_l)$ and ψ is a Boolean combination of atomic formulas of the form $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \geq c$, or $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \leq c$, or $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) = c$, or $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \equiv r \bmod p$, satisfying that $\Delta \subseteq \Sigma \times 2^{\{R_1, \dots, R_l\}}$, $c \in \mathbb{N}$, $p \geq 2$, $0 \leq r < p$, and if $\tau = (\sigma, \mathcal{R})$, then $\tau(y) = \sigma(y) \wedge \bigwedge\limits_{i: 1 \leq i \leq l} \eta_{R_i}(y)$, where $\eta_{R_i}(y) = R_i(y)$ if $R_i \in \mathcal{R}$, and $\eta_{R_i}(y) = \neg R_i(y)$ otherwise.

The semantics of $EMSO^2(+1, \sim, \Sigma)$ formulas can be extended naturally to $EMSO^2_{\#}(+1, \sim, \Sigma)$ formulas by interpreting formulas $\forall x \psi$ as the counting constraints for each class. Let's take the formula $\forall x \left(\#_{x \sim y \wedge \tau(y)}(y) \geq c\right)$ as an example: Given a data word w over the alphabet $\Sigma \cup 2^{\{R_1, \dots, R_l\}}$, $w \models \forall x \left(\#_{x \sim y \wedge \tau(y)}(y) \geq c\right)$ iff for each class X of w, the number of τ-positions in X is at least c.

▶ **Theorem 6.** *$EMSO^2_{\#}(+1, \sim, \Sigma)$ and CDA are expressively equivalent.*

- *Given a $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$, a CDA $\mathscr{D} = (\mathscr{A}, \varphi')$ of doubly exponential size can be constructed such that $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\exists R_1 \dots R_l(\varphi \wedge \forall x \psi))$. In addition, the size of the output alphabet of $\mathscr{A}$ is at most exponential over the size of $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$.*
- *Given a CDA $\mathscr{D} = (\mathscr{A}, \varphi)$, a $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula φ' of polynomial size can be constructed such that $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\varphi')$.*

4 The nonemptiness problem of CDA

▶ **Theorem 7.** *The nonemptiness of CDA can be decided in 3-NEXPTIME.*

The rest of this section is devoted to the proof of Theorem 7. Although the structure of the proof is similar to that for WDA in [11, 5], the proofs of several lemmas become more complicated.

Through this section, let $\mathscr{D} = (\mathscr{A}, \varphi)$ be a commutative data automaton such that $\mathscr{A} = (Q, \Sigma \times \{\bot, \top\}, \Gamma, \delta, q_0, F)$ and φ is a QFSP formula over the variable set V_Γ.

Because we are concerned with the nonemptiness problem, without loss of generality, we can assume that $\mathscr{A} = (Q, \Gamma \times \{\bot, \top\}, \delta, q_0, F)$ is just a finite automaton over the alphabet $\Gamma \times \{\bot, \top\}$. Then the nonemptiness of $\mathscr{D}$ is reduced to the following problem.

PROBLEM:	NONEMPTINESS-PROFILE
INPUT:	A finite automaton $\mathscr{A} = (Q, \Gamma \times \{\bot, \top\}, \delta, q_0, F)$ and a QFSP formula φ over V_Γ
QUESTION:	is there a data word w over Γ accepted by $(\mathscr{A}, \varphi)$, i.e. $\mathsf{Profile}(w)$ is accepted by $\mathscr{A}$ and $w \models_c \varphi$?

The outline of the proof goes as follows.

- At first, a finite automaton $\mathscr{A}'$ of exponential size over the alphabet Γ', and a QFSP formula φ' in the normal form of doubly exponential size over the variable set $V_{\Gamma'}$, are constructed from $\mathscr{D} = (\mathscr{A}, \varphi)$ such that the problem of NONEMPTINESS-PROFILE is reduced to the following problem,

  "is there a *locally different* data word w over the alphabet Γ' such that $\mathsf{Proj}(w)$ is accepted by $\mathscr{A}'$ and $w \models_c \varphi'$?"

  We would like to point out that the finite automaton $\mathscr{A}'$ runs directly on the *projections* of data words, instead of the profile words of them.

  Let's call this problem NONEMPTINESS-LOCALLY-DIFFERENT, which is formally defined as follows.

PROBLEM:	NONEMPTINESS-LOCALLY-DIFFERENT
INPUT:	A finite automaton $\mathscr{A} = (Q, \Gamma, \delta, q_0, F)$ and a QFSP formula φ over V_Γ in the normal form
QUESTION:	is there a locally different data word w over Γ accepted by $(\mathscr{A}, \varphi)$, i.e. $\mathsf{Proj}(w)$ is accepted by $\mathscr{A}$ and $w \models_c \varphi$?

- Then a 2-NEXPTIME algorithm is presented to solve the problem of NONEMPTINESS-LOCALLY-DIFFERENT.

From the above description of the proof outline, it is evident that NONEMPTINESS-PROFILE can be decided in 4-NEXPTIME. By a finer analysis, the complexity can be shown in 3-NEXPTIME.

Since the reduction of the problem of NONEMPTINESS-PROFILE to the problem of NONEMPTINESS-LOCALLY-DIFFERENT completely mimics that for WDA In [11, 5], it is omitted here due to the lack of space.

In the rest of this section, we will focus on the problem of NONEMPTINESS-LOCALLY-DIFFERENT. Before presenting an algorithm to solve the problem, we will state and prove two lemmas.

4.1 Two lemmas

We first introduce some notations.

▶ **Definition 8.** *Let* $\varphi = \bigvee\limits_{1 \leq i \leq m} \varphi_i$ *be a QFSP formula in the normal form over the variable set* V_Γ*,* p_0 *be the normalization number of* φ*, and for every* $i : 1 \leq i \leq m$*,* $\varphi_i = \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$*, where*

$\varphi_{i,\gamma}$ is either $x_\gamma = c_{i,\gamma}$ or $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$ for some $c_{i,\gamma}, r_{i,\gamma} : 0 \leq c_{i,\gamma}, r_{i,\gamma} < p_0$. Then for each $\gamma \in \Gamma$, define two subsets of $\{1, \ldots, m\}$, denoted by $I_E(\varphi, \gamma)$ and $I_M(\varphi, \gamma)$, as follows: For every $i : 1 \leq i \leq m$,

- *$i \in I_E(\varphi, \gamma)$ iff $\varphi_{i,\gamma}$ is $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} > 0$,*
- *$i \in I_M(\varphi, \gamma)$ iff $\varphi_{i,\gamma}$ is $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$.*

Note that if $i \notin I_E(\varphi, \gamma) \cup I_M(\varphi, \gamma)$, then it holds that $\varphi_{i,\gamma}$ is $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} = 0$.

▶ **Definition 9.** *Let w be a data word over Γ and $\varphi = \bigvee\limits_{1 \leq i \leq m} \varphi_i$ be a QFSP formula over the variable set V_Γ in the normal form. If $w \models_c \varphi$, then for each data value d occurring in w, there is a unique $i : 1 \leq i \leq m$ such that $\mathsf{Proj}(w|_X) \models \varphi_i[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$, where X is the class of w corresponding to d. This unique number i is called* the index of the class condition *φ for d, denoted by $\mathsf{idx}_\varphi(d)$.*

We are ready to state and prove the two lemmas.

▶ **Lemma 10.** *For every QFSP formula $\varphi = \bigvee\limits_{1 \leq i \leq m} \varphi_i$ over the variable set V_Γ in the normal form, there is an EP formula $\psi = \exists y_1 \ldots \exists y_m \psi'$ of polynomial size such that for each word v over Γ, $v \models \psi[\mathsf{Parikh}(v)]$ iff there is a data word w such that $\mathsf{Proj}(w) = v$ and $w \models_c \varphi$.*

Proof. Suppose φ is a QFSP formula in the normal form over the variable set V_Γ with the normalization number p_0. Then $\varphi = \bigvee\limits_{i : 1 \leq i \leq m} \varphi_i$, where φ_i is of the form $\bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$ such that $\varphi_{i,\gamma}$ is equal to $x_\gamma = c_{i,\gamma}$ or $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$ for some $c_{i,\gamma}, r_{i,\gamma} : 0 \leq c_{i,\gamma}, r_{i,\gamma} < p_0$. In addition, those φ_i's are mutually exclusive.

Let $\psi = \exists y_1 \ldots \exists y_m \psi'$ such that ψ' is a conjunction of the quantifier-free Presburger formulas ψ'_1, ψ'_2, and ψ'_3, where

1. $\psi'_1 := \bigwedge\limits_{\gamma \in \Gamma} \left(x_\gamma - \left(\sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} y_i \right) - \left(\sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}) y_i \right) \geq 0 \right)$. In particular, if $I_E(\varphi, \gamma) = \emptyset$, then $\sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} y_i$ is replaced by 0 in ψ'_1. Similarly, if $I_M(\varphi, \gamma) = \emptyset$, then $\sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}) y_i$ is replaced by 0 in ψ'_1.
2. $\psi'_2 := \bigwedge\limits_{\gamma \in \Gamma} \left(\left(\bigwedge\limits_{i \in I_M(\varphi,\gamma)} y_i = 0 \right) \to x_\gamma - \left(\sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} y_i \right) = 0 \right)$. In particular, if $I_M(\varphi, \gamma) = \emptyset$, then $\left(\bigwedge\limits_{i \in I_M(\varphi,\gamma)} y_i = 0 \right)$ is replaced by *true*; on the other hand, if $I_E(\varphi, \gamma) = \emptyset$, then $\sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} y_i$ is replaced by 0.
3. $\psi'_3 := \bigwedge\limits_{\gamma \in \Gamma} \left(x_\gamma - \left(\sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} y_i + \sum\limits_{i \in I_M(\varphi,\gamma)} r_{i,\gamma} y_i \right) \equiv 0 \bmod p_0 \right)$.

"If" part:

Suppose there is a data word w such that $\mathsf{Proj}(w) = v$ and $w \models_c \varphi$, namely, for each class X in w, $\mathsf{Proj}(w|_X) \models \varphi[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$.

For each $i : 1 \leq i \leq m$, let D_i be the set of data values d occurring in w such that $\mathsf{idx}_\varphi(d) = i$. Note that $(D_i)_{1 \leq i \leq m}$ forms a partition of the set of all the data values occurring in w. In addition, let $k_i = |D_i|$ for each $i : 1 \leq i \leq m$.

It is sufficient to verify that $v \models \psi'[\mathsf{Parikh}(v), \overline{k}]$ in order to show $v \models \psi[\mathsf{Parikh}(v)]$.

Let's exemplify the argument by demonstrating that $v \models \psi'_2[\mathsf{Parikh}(v), \overline{k}]$.

Suppose $k_i = 0$ for each $i \in I_M(\varphi, \gamma)$. Then $D_i = \emptyset$ for each $i \in I_M(\varphi, \gamma)$. We want to show that $\#_v(\gamma) = \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i$.

For each data value $d \in D_i$ such that $i \notin I_M(\varphi, \gamma)$,

- if $i \in I_E(\varphi, \gamma)$, i.e. $\varphi_{i,\gamma}$ is equal to $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} > 0$, then the letter γ occurs exactly $c_{i,\gamma}$ times in the class of w corresponding to d;
- if $i \notin I_E(\varphi, \gamma) \cup I_M(\varphi, \gamma)$, i.e. $\varphi_{i,\gamma}$ is equal to $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} = 0$, then the letter γ does not occur in the class of w corresponding to d.

Since $(D_i)_{1 \leq i \leq m}$ is a partition of the set of all the data values occurring in w, it follows that $\#_v(\gamma) = \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i$. So $v \models \psi_2'[\mathsf{Parikh}(v), \overline{k}]$.

"Only if" part:

Suppose $v \models \psi[\mathsf{Parikh}(v)]$. Then there are a tuple of numbers $\overline{k} = k_1, \dots, k_m$ such that $v \models \psi'[\mathsf{Parikh}(v), \overline{k}]$.

Let $K = k_1 + \cdots + k_m$. Define a function $\xi : \{1, \dots, K\} \rightarrow \{1, \dots, m\}$ such that $|\xi^{-1}(i)| = k_i$ for each $i : 1 \leq i \leq m$.

In the following, we assign the data values from $\{1, \dots, K\}$ to the positions in v to get a data word w such that $w \models_c \varphi$, namely, for each class X of w, $\mathsf{Proj}(w|_X) \models \varphi[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$.

From the fact that $v \models \psi_1'[\mathsf{Parikh}(v), \overline{k}]$, we know that for each $\gamma \in \Gamma$,

$$\#_v(\gamma) \geq \sum_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i \; + \sum_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}) k_i.$$

We assign the data values in $\{1, \dots, K\}$ to the positions in v through the following two-step procedure.

Step 1 For every $\gamma \in \Gamma$ and every $i : 1 \leq i \leq m$, assign the data values in $\xi^{-1}(i)$ to the γ-positions in v such that each data value in $\xi^{-1}(i)$ is assigned to exactly $(c_{i,\gamma})$ γ-positions if $i \in I_E(\varphi, \gamma)$, and is assigned to exactly $(p_0 + r_{i,\gamma})$ γ-positions if $i \in I_M(\varphi, \gamma)$.

Step 2 For every $\gamma \in \Gamma$ such that there is $i \in I_M(\varphi, \gamma)$ satisfying that $k_i > 0$, select such an index i and a data value from $\xi^{-1}(i)$, denoted by d_γ, and assign d_γ to all the γ-positions which have not been assigned data values after Step 1.

Now all the positions of v have been assigned data values from $\{1, \dots, K\}$, let w be the resulting data word.

For every $\gamma \in \Gamma$, if there are still γ-positions that have not been assigned data values after Step 1, then $\#_v(\gamma) > \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i \; + \sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}) k_i \geq \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i$. From the fact that $v \models \psi_2'[\mathsf{Parikh}(v), \overline{k}]$, it follows that there is $i \in I_M(\varphi, \gamma)$ such that $k_i > 0$. So a data value d_γ can be selected and assigned to all the pending γ-positions in Step 2.

It remains to show that $w \models_c \varphi$. It is sufficient to prove that for every $i : 1 \leq i \leq m$ and every data value $d \in \xi^{-1}(i)$, $\mathsf{Proj}(w|_X) \models \varphi_i[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$, where X is the class of w corresponding to d. Because $\mathsf{Proj}(w|_X) = v|_X$ and $\varphi_i = \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$, it is equivalent to show that for every $i : 1 \leq i \leq m$, $d \in \xi^{-1}(i)$, and $\gamma \in \Gamma$, we have $v|_X \models \varphi_{i,\gamma}[\#_{v|_X}(\gamma)]$, where X is the class of w corresponding to d.

Suppose $i : 1 \leq i \leq m$, $d \in \xi^{-1}(i)$, and $\gamma \in \Gamma$. Let X be the class of w corresponding to d. In the following, we show that $v|_X \models \varphi_{i,\gamma}[\#_{v|_X}(\gamma)]$.

From the data value assignment procedure, we know that there are still $\big(\#_v(\gamma) - \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i \; - \sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}) k_i\big)$ γ-positions which have not been assigned data values after Step 1. Because $v \models \psi_3'[\mathsf{Parikh}(v), \overline{k}]$, it follows that $\#_v(\gamma) - \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i \; -$

$\sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma})k_i \equiv 0 \bmod p_0$. So there is $t_\gamma \in \mathbb{N}$ such that $\#_v(\gamma) - \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i - \sum\limits_{i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma})k_i = t_\gamma p_0$.

We distinguish between the following three cases.

Case $i \in I_E(\varphi, \gamma)$. Then $\varphi_{i,\gamma}$ is $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} > 0$. From the data value assignment procedure, we know that each data value in $\xi^{-1}(i)$, including d, has been assigned to exactly $(c_{i,\gamma})$ γ-positions. This implies that $\#_{v|_X}(\gamma) = c_{i,\gamma}$. So $v|_X \models \varphi_{i,\gamma}[\#_{v|_X}(\gamma)]$.

Case $i \in I_M(\varphi, \gamma)$. Then $\varphi_{i,\gamma}$ is $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$. From the data value assignment procedure, we know that the data value d is assigned to $(p_0 + r_{i,\gamma})$ γ-positions if $d \neq d_\gamma$, and assigned to $(p_0 + r_{i,\gamma} + t_\gamma p_0)$ γ-positions otherwise. Therefore, $\#_{v|_X}(\gamma) = p_0 + r_{i,\gamma}$ or $p_0 + r_{i,\gamma} + t_\gamma p_0$. It follows that $v|_X \models \varphi_{i,\gamma}[\#_{v|_X}(\gamma)]$.

Case $i \notin I_E(\varphi, \gamma) \cup I_M(\varphi, \gamma)$. Then $\varphi_{i,\gamma}$ is $x_\gamma = c_{i,\gamma}$ and $c_{i,\gamma} = 0$. From the data value assignment procedure, we know that each data value in $\xi^{-1}(i)$, including d, has not been assigned to any γ-position in v. Therefore, $\#_{v|_X}(\gamma) = 0$ and $v|_X \models \varphi_{i,\gamma}[\#_{v|_X}(\gamma)]$. ◀

▶ **Definition 11.** *Let $\varphi = \bigvee\limits_{1 \leq i \leq m} \varphi_i$ be a QFSP formula in the normal form over the variable set V_Γ with the normalization number p_0 such that for each $i : 1 \leq i \leq m$, $\varphi_i = \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$, where $\varphi_{i,\gamma}$ is $x_\gamma = c_{i,\gamma}$ or $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$ for $0 \leq c_{i,\gamma}, r_{i,\gamma} < p_0$. Moreover, for each $i : 1 \leq i \leq m$, let*

$$h_i = \sum_{\gamma : i \in I_E(\varphi,\gamma)} c_{i,\gamma} + \sum_{\gamma : i \in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma}).$$

Let w be a data word over the alphabet Γ, then w is said to satisfy the class condition φ with "many" data values if $w \models_c \varphi$ and for each $i : 1 \leq i \leq m$, either $k_i = 0$ or $k_i \geq \max(2p_0 + 1,\ 2h_i + 3)$, where k_i is the number of data values d occurring in w such that $\mathsf{idx}_\varphi(d) = i$.

Let $v \in \Gamma^*$ and $\psi = \exists y_1 \ldots \exists y_m \psi'$ be the EP formula obtained from φ as in Lemma 10. Then v is said to satisfy ψ with "large" numbers if there are a tuple of numbers $\overline{k} = k_1, \ldots, k_m$ such that $v \models \psi'[\mathsf{Parikh}(v), \overline{k}]$ and for each $i : 1 \leq i \leq m$, either $k_i = 0$ or $k_i \geq \max(2p_0 + 1,\ 2h_i + 3)$.

▶ **Lemma 12.** *Let $\varphi = \bigvee\limits_{1 \leq i \leq m} \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$ be a QFSP formula in the normal form with the normalization number p_0. Moreover, let $\psi = \exists y_1 \ldots \exists y_m \psi'$ be the EP formula obtained from φ as stated in Lemma 10. Then for any $v \in \Gamma^*$, $v \models \psi$ with large numbers iff there is a locally different data word w such that $\mathsf{Proj}(w) = v$ and $w \models_c \varphi$ with many data values.*

Proof. "If" part: Obvious.

"Only if"part:

Suppose v satisfies ψ with large numbers, i.e. there are numbers $\overline{k} = k_1, \ldots, k_m$ such that $v \models \psi'[\mathsf{Parikh}(v), \overline{k}]$ and for each $i : 1 \leq i \leq m$, either $k_i = 0$ or $k_i \geq \max(2p_0 + 1, 2h_i + 3)$.

Let $K = k_1 + \cdots + k_m$. Define a function $\xi : \{1, \ldots, K\} \to \{1, \ldots, m\}$ such that $|\xi^{-1}(i)| = k_i$ for each $i : 1 \leq i \leq m$.

As in the proof of Lemma 10, we assign data values in $\{1, \ldots, K\}$ to the positions of v to get a desired data word w. The assignment procedure is divided into two steps, Step 1 and Step 2.

Step 1:

The same as Step 1 of the data value assignment procedure in the proof of the "Only if" part of Lemma 10.

After Step 1, we get a partial data word where some positions still have no data values. Let's assign a special data value, say $\sharp$, to all those positions without data values, then we get a data word $w_1 = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2}\dots\binom{\gamma_n}{d_n}$.

In w_1, there may exist positions j such that $d_j = d_{j+1}$ and $d_j, d_{j+1} \neq \sharp$. Let's call these positions as *conflicting* positions of w_1.

Let j be a conflicting position of w_1, $a = \gamma_j$, and $i : 1 \leq i \leq m$ such that $d_j \in \xi^{-1}(i)$. From the description of Step 1, we know that d_j occurs exactly $h_i = \sum\limits_{\gamma: i \in I_E(\varphi,\gamma)} c_{i,\gamma} + \sum\limits_{\gamma: i \in I_M(\gamma)} (p_0 + r_{i,\gamma})$ times in w_1. It follows that there are at most $2h_i$ positions adjacent to a position with the data value d_j. Since $k_i \geq \max(2p_0+1, 2h_i+3)$, it follows that there are (at least) *three* positions j'_1, j'_2, j'_3 such that $d_{j'_1}, d_{j'_2}, d_{j'_3} \in \xi^{-1}(i)$, $d_{j'_1}, d_{j'_2}, d_{j'_3}$ are pairwise distinct, $\gamma_{j'_1} = \gamma_{j'_2} = \gamma_{j'_3} = a$, and $d_{j'_1-1}, d_{j'_2-1}, d_{j'_3-1}, d_{j'_1+1}, d_{j'_2+1}, d_{j'_3+1} \neq d_j$. From this, we deduce that there is a position j' such that $\gamma_{j'} = a$, $d_{j'} \in \xi^{-1}(i)$, $d_{j'} \neq d_{j-1}, d_j$, and $d_j \neq d_{j'-1}, d_{j'+1}$. Because $d_{j'} \neq d_{j-1}, d_{j+1}$ ($d_{j+1} = d_j$ since j is conflicting) and $d_j \neq d_{j'-1}, d_{j'+1}$, we can swap the data value d_j in the position j and the data value $d_{j'}$ in the position j' to make the two positions j and j' non-conflicting. Let w'_1 be the data word after the swapping. It follows that w'_1 has less conflicting positions than w_1.

Continue like this, we finally get a data word w''_1 without conflicting positions.

But w''_1 may still contain the special data value $\sharp$. If this is the case, then from the description of Step 1, we know that there exists at least one $\gamma \in \Gamma$ such that $\#_v(\gamma) > \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i + \sum\limits_{i \in I_M(\gamma)} (p_0 + r_{i,\gamma}) k_i$.

Let $\gamma \in \Gamma$ such that $\#_v(\gamma) > \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i + \sum\limits_{i \in I_M(\gamma)} (p_0 + r_{i,\gamma}) k_i$.

From the fact that $v \models \psi'_3[\mathsf{Parikh}(v), \overline{k}]$, it follows that $\#_v(\gamma) - \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i - \sum\limits_{i \in I_M(\gamma)} (p_0 + r_{i,\gamma}) k_i \equiv 0 \bmod p_0$. So there is $t_\gamma \geq 1$ such that $\#_v(\gamma) - \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i - \sum\limits_{i \in I_M(\gamma)} (p_0 + r_{i,\gamma}) k_i = t_\gamma p_0$. Therefore, there are $(t_\gamma p_0)$ γ-positions in w_1 with the data value $\sharp$. Because w_1 and w''_1 have the same set of positions with the data value $\sharp$, it follows that there are also $(t_\gamma p_0)$ γ-positions in w''_1 with the data value $\sharp$. Let $j_{\gamma,1} < \dots < j_{\gamma, t_\gamma p_0}$ be a list of all such γ-positions in w''_1.

On the other hand, because $v \models \psi'_2[\mathsf{Parikh}(v), \overline{k}]$ and $\#_v(\gamma) > \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i$, it follows that there is $i \in I_M(\varphi, \gamma)$ such that $k_i > 0$. Let i_γ be such an index i. Then from the assumption that v satisfies ψ with large numbers, we know that $k_{i_\gamma} \geq \max(2p_0+1, 2h_{i_\gamma}+3)$.

Step 2:

For each $\gamma \in \Gamma$ such that $\#_v(\gamma) > \sum\limits_{i \in I_E(\varphi,\gamma)} c_{i,\gamma} k_i + \sum\limits_{i \in I_M(\gamma)} (p_0 + r_{i,\gamma}) k_i$, assign the data values from $\{1, \dots, K\}$ to the γ-positions with the data value $\sharp$ in w''_1 as follows. We distinguish between the following two cases.

- *Case $t_\gamma \geq 2$.*

  Initially set $s := 1$. Repeat following procedure until $s > t_\gamma$.

  Let $J = \{j_{\gamma,s}, j_{\gamma,t_\gamma+s}, \dots, j_{\gamma,t_\gamma(p_0-1)+s}\}$ and J' be the set of all positions adjacent to a position in J. In addition, let D be the set of data values (except $\sharp$) occurring in the positions belonging to J'. Because $|J| = p_0$, we have $|D| \leq 2p_0$. On the other hand, $k_{i_\gamma} \geq 2p_0 + 1$, it follows that there is $d \in \xi^{-1}(i_\gamma) \setminus D$.

  Assign the data value d to all the positions in J. Then we still get a non-conflicting data word, since all the positions in J are not adjacent to each

other.
Set $s := s+1$.

- *Case $t_\gamma = 1$.*
  Let $J = \{j_{\gamma,1}, j_{\gamma,2}, \dots, j_{\gamma,p_0}\}$ and J' be the set of all positions adjacent to a position in J. In addition, let D be the set of data values (except $\sharp$) occurring in the positions belonging to J'. Because $|J| = p_0$, we have $|D| \leq 2p_0$. On the other hand, $k_{i_\gamma} \geq 2p_0 + 1$, it follows that there is $d \in \xi^{-1}(i_\gamma) \setminus D$.
  Because $i_\gamma \in I_M(\varphi, \gamma)$, each data value in $\xi^{-1}(i_\gamma)$ has been assigned to exactly $(p_0 + r_{i_\gamma,\gamma})$ γ-positions in Step 1. During Step 1, we can do the assignments in a way so that all the positions in J, i.e. the p_0 γ-positions without data value, are not adjacent to each other. Therefore, we can assign the data value d to every position in J and still get a non-conflicting data word.

Let w be the resulting data word after the two steps of data value assignments. Then w is locally different. Similar to the proof of the "Only if" part of Lemma 10, we can show that for each $i : 1 \leq i \leq m$ and each data value $d \in \xi^{-1}(i)$, $\mathsf{Proj}(w|_X) \models_c \varphi_i[\mathsf{Parikh}(\mathsf{Proj}(w|_X))]$, where X is the class of w corresponding to d. From this, it follows that for each $i : 1 \leq i \leq m$, the number of data values in w such that $i \in \mathsf{idx}_\varphi(d)$ is equal to k_i. Since for each $i : 1 \leq i \leq m$, either $k_i = 0$ or $k_i \geq \max(2p_0 + 1, 2h_i + 3)$, we conclude that $w \models_c \varphi$ with many data values. ◀

4.2 Algorithm for NONEMPTINESS-LOCALLY-DIFFERENT

We first give an algorithm for the following problem.

PROBLEM:	NONEMPTINESS-LOCALLY-DIFFERENT-MANY
INPUT:	A finite automaton $\mathscr{A} = (Q, \Gamma, \delta, q_0, F)$ and a QFSP formula φ over V_Γ in the normal form
QUESTION:	is there a locally different data word w over the alphabet Γ such that $\mathsf{Proj}(w)$ is accepted by $\mathscr{A}$ and $w \models_c \varphi$ with many data values?

From Lemma 12, it follows that NONEMPTINESS-LOCALLY-DIFFERENT-MANY can be solved by the following algorithm.

Suppose the normalization number of φ is p_0 and $\varphi = \bigvee\limits_{i:1\leq i\leq m} \bigwedge\limits_{\gamma\in\Gamma} \varphi_{i,\gamma}$ such that each $\varphi_{i,\gamma}$ is either $x_\gamma = c_{i,\gamma}$ or $x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$ for some $c_{i,\gamma}, r_{i,\gamma} : 0 \leq c_{i,\gamma}, r_{i,\gamma} < p_0$. Let $\psi = \exists y_1 \dots \exists y_m \psi'$ be the existential Presburger formula obtained from φ as stated in Lemma 10. For every $i : 1 \leq i \leq m$, let $h_i = \sum\limits_{\gamma: i \in I_E(\varphi,\gamma)} c_{i,\gamma} + \sum\limits_{\gamma: i\in I_M(\varphi,\gamma)} (p_0 + r_{i,\gamma})$.

1. *Construct the following EP formula ψ_g,*

$$\psi_g := \exists y_1 \dots \exists y_m \left(\psi' \wedge \bigwedge_{1\leq i\leq m} (y_i = 0 \vee (y_i \geq 2p_0 + 1 \wedge y_i \geq 2h_i + 3)) \right).$$

2. *Decide the nonemptiness of the Presburger automaton $(\mathscr{A}, \psi_g)$.*

Now we consider the problem of NONEMPTINESS-LOCALLY-DIFFERENT.

For a data word $w \in (\Gamma \times \mathbb{D})^*$, if $w \models_c \varphi$, then for each $i : 1 \leq i \leq m$, let k_i be the number of data values d occurring in w such that $\mathsf{idx}_\varphi(d) = i$. For each $i : 1 \leq i \leq m$ such that

$k_i < \max(2p_0+1, 2h_i+3)$, if we take the k_i data values d such that $\mathsf{idx}_\varphi(d) = i$ as constants, then the problem of NONEMPTINESS-LOCALLY-DIFFERENT can be solved similar to the problem of NONEMPTINESS-LOCALLY-DIFFERENT-MANY. More specifically, the algorithm goes as follows.

1. *Guess a set $J \subseteq \{1, \dots, m\}$ and sets of constants D_j's.*
   a) *Guess a set $J \subseteq \{1, \dots, m\}$.*
   b) *For each $j \in J$, guess an integer $s_j < \max(2p_0+1, 2h_i+3)$.*
   c) *For each $j \in J$, fix a set $D_j = \{\alpha^j_1, \dots, \alpha^j_{s_j}\}$ of constants such that D_j's are mutually disjoint and $D_j \cap \mathbb{D} = \emptyset$. Let $D_J = \cup_{j \in J} D_j$.*
2. *Construct an automaton $\mathscr{A}'$ over the alphabet $\Gamma \cup \Gamma \times D_J$ from $(\mathscr{A}, \varphi)$ such that $\mathscr{A}'$ accepts a word $v = \lambda_1 \dots \lambda_n \in (\Gamma \cup \Gamma \times D_J)^*$ iff the following conditions hold.*
   - *A symbol (γ, d) appears in v iff there exists $j \in J$ such that $j \in I_E(\varphi, \gamma) \cup I_M(\varphi, \gamma)$ and $d \in D_j$.*
   - *Let $u = \gamma_1 \dots \gamma_n \in \Gamma^*$ such that*
   $$\gamma_i = \begin{cases} \lambda_i & \text{if } \lambda_i \in \Gamma, \\ \gamma & \text{if } \lambda_i = (\gamma, d) \in \Gamma \times D_J. \end{cases}$$
   Then u is accepted by $\mathscr{A}$.
   - *For any $i : 1 \leq i < n$, if $\lambda_i = (\gamma, d)$ and $\lambda_{i+1} = (\gamma', d')$, then $d \neq d'$.*
   - *For any $j \in J$ and any $\gamma \in \Gamma$, the following holds: If $j \in I_E(\varphi, \gamma)$, then for each $d \in D_j$, the letter (γ, d) occurs exactly $c_{j,\gamma}$ times in v. If $j \in I_M(\varphi, \gamma)$, then for each $d \in D_j$, the number of occurrences of the letter (γ, d) is at least p_0 and equal to $r_{i,\gamma}$ modulo p_0.*
3. *Construct the following EP formula $\psi_{g,J}$,*
   $$\psi_{g,J} = \exists y_1 \dots \exists y_m \left(\psi' \wedge \bigwedge_{i \in J} y_i = 0 \wedge \bigwedge_{i \notin J} (y_i \geq 2p_0 + 1 \wedge y_i \geq 2h_i + 3) \right).$$
   Note that $\psi_{g,J}$ is a EP formula with free variables from V_Γ, and contains no variables $x_{(\gamma,d)}$ with $(\gamma, d) \in \Gamma \times D_J$.
4. *Decide the nonemptiness of the Presburger automaton $(\mathscr{A}', \psi_{g,J})$.*

The proof of the correctness of the above algorithm for NONEMPTINESS-LOCALLY-DIFFERENT follows the same line as the proof for SAT-LOCALLY-DIFFERENT in [5].

5 Commutative Büchi data automata

In this section, we consider data automata with commutative class conditions over data ω-words.

Let $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$ with the linear order ($<$) and the addition ($+$) operation of $\mathbb{N}$ extended in a natural way, i.e. $n < \omega$ for any $n \in \mathbb{N}$, and $\omega + n = \omega$ for any $n \in \mathbb{N}_\omega$.

The definition of the Parikh images of finite words can be easily extended to ω-words: Given an ω-word v over an alphabet $\Gamma = \{\gamma_1, \dots, \gamma_l\}$, $\mathsf{Parikh}(v) = (\#_v(\gamma_1), \dots, \#_v(\gamma_l))$, where for each $i : 1 \leq i \leq l$, $\#_v(\gamma_i)$ is still the number of occurrences of γ_i in v, in particular, if γ_i occurs infinitely many times in v, then $\#_v(\gamma_i) = \omega$.

Similar to QFSP formulas, we define ω-QFSP formulas over a variable set X as follows.

The syntax of ω-QFSP formulas is the same as QFSP formulas, except that the atomic formulas can also be of the form $x = \omega$ (where $x \in X$).

The ω-QFSP formulas are interpreted on $\mathbb{N}_\omega$: Let $\pi : X \to \mathbb{N}_\omega$, then the atomic ω-QFSP formulas are interpreted as follows,

- $\pi \models x_1 + \cdots + x_n \text{ op } c$ if $\pi(x_1) + \cdots + \pi(x_n) \text{ op } c$, where $\text{op} \in \{\leq, \geq, =\}$,
- $\pi \models x_1 + \cdots + x_n \equiv r \bmod p$ if $\pi(x_1) + \cdots + \pi(x_n) < \omega$ and $\pi(x_1) + \cdots + \pi(x_n) \equiv r \bmod p$,
- $\pi \models x = \omega$ if $\pi(x) = \omega$;

in addition, the Boolean operators are interpreted in a standard way.

Similar to Proposition 3, there is a normal form for ω-QFSP formulas.

▶ Proposition 13. *Let $\varphi(x_1, \ldots, x_k)$ be a ω-QFSP formula. Then there exists an exponential-time algorithm to transform φ into a ω-QFSP formula $\bigvee\limits_{i:1 \leq i \leq m} \varphi_i$ of size $2^{O(k|\varphi|)}$ such that there is $p_0 : 2 \leq p_0 \leq 2^{|\varphi|}$ satisfying that*

- *each φ_i is of the form $\bigwedge\limits_{1 \leq j \leq k} \varphi_{i,j}$;*
- *for each $j : 1 \leq j \leq k$, $\varphi_{i,j}$ is equal to $x_j = c_{i,j}$ or $x_j \geq p_0 \wedge x_j \equiv r_{i,j} \bmod p_0$, or $x_j = \omega$ for $c_{i,j}, r_{ij} : 0 \leq c_{i,j}, r_{i,j} < p_0$;*
- *in addition, those φ_i's are mutually exclusive.*

Let w be a data ω-word over an alphabet Γ and φ be a ω-QFSP formula over the variable set V_Γ, then the definition of $w \models_c \varphi$, i.e. w satisfies the class condition φ, is a natural extension of that for data words.

A *commutative Büchi data automaton* (CBDA) is a binary tuple $(\mathscr{A}, \varphi)$, where $\mathscr{A} = (Q, \Sigma \times \{\perp, \top\}, \Gamma, \delta, q_0, F)$ is a Büchi letter-to-letter transducer and φ is a ω-QFSP formula over the variable set V_Γ.

A CBDA $(\mathscr{A}, \varphi)$ accepts a data ω-word $w = \binom{\sigma_1}{d_1}\binom{\sigma_2}{d_2}\cdots\binom{\sigma_n}{d_n}$ if there is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(w)$ which produces a ω-word $\gamma_1\gamma_2\ldots$ such that the data ω-word $w' = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2}\ldots\ldots$ satisfies that $w' \models_c \varphi$.

Similar to the logic $EMSO^2_\#(+1, \sim, \Sigma)$ in Section 3, we define the logic $E_\infty MSO^2_\#(+1, \sim, \Sigma)$ as follows: It includes all the formulas $\exists_\infty R_1 \ldots \exists_\infty R_k \exists S_1 \ldots \exists S_l (\varphi \wedge \forall x \psi)$, where $\varphi \in FO^2(+1, \sim, \Sigma, R_1, \ldots, R_k, S_1, \ldots, S_l)$ and ψ is the same as the ψ in $EMSO^2_\#(+1, \sim, \Sigma)$ formulas, except that the atomic formulas in ψ can be also of the form $\#_{x \sim y \wedge \tau(y)}(y) = \omega$.

The semantics of $E_\infty MSO^2(+1, \sim, \Sigma)$ formulas are defined similar to $EMSO^2_\#(+1, \sim, \Sigma)$ formulas, except that the unary relation symbols $R_1, \ldots, R_k$ are restricted to bind to infinite sets and $\#_{x \sim y \wedge \tau(y)}(y) = \omega$ are interpreted as the fact that the symbol τ appears infinitely many times in the class that contains the position x.

Similar to CDA, we also have the following logical characterization of CBDA.

▶ **Theorem 14.** *$E_\infty MSO^2_\#(+1, \sim, \Sigma)$ and CBDA are expressively equivalent.*

The proof of Theorem 14 is similar to that for WBDA in [11].

▶ **Theorem 15.** *The nonemptiness of CBDA can be decided in 4-NEXPTIME.*

The proof of Theorem 15 is by a nondeterminstic exponential time reduction to the nonemptiness of CDA on data words.

6 Conclusions and future work

In this paper, aiming at finding a formalism which achieves a good balance between expressibility and complexity, we introduced a natural restriction of data automata, called commutative data automata, which are data automata with commutative class conditions. We demonstrated that while the expressive power of commutative data automata is strictly stronger than weak data automata, it still preserves the virtue of weak data automata, namely, the elementary complexity for the nonemptiness problem. In addition, we defined commutative Büchi data automata and extended the results to data ω-words.

There are several directions for the future work.

- At first, we plan to investigate the lower bound for the nonemptiness of commutative data automata.
- It is also interesting to investigate whether the commutative class conditions can be relaxed to partially commutative languages, while still preserving the elementary complexity.
- Finally, it is interesting to investigate whether similar models can be defined for data trees.

Acknowledgements. The author thanks Anca Muscholl for the comments and suggestions on a earlier draft of this paper.

References

1 Mikoaj Bojańczyk and Sławomir Lasota. An extension of data automata that captures XPath. In *LICS '10*, pages 243–252, 2010.

2 Mikoaj Bojańczyk, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data trees and XML reasoning. *J. ACM*, 56(3):1–48, 2009.

3 Mikolaj Bojańczyk, Anca Muscholl, Thomas Schwentick, Luc Segoufin, and Claire David. Two-variable logic on words with data. In *LICS '06*, pages 7–16, 2006.

4 M Chrobak. Finite automata and unary languages. *Theor. Comput. Sci.*, 47(2):149–158, 1986.

5 Claire David, Leonid Libkin, and Tony Tan. On the satisfiability of two-variable logic over data words. In *LPAR'10*, pages 248–262, 2010.

6 Stéphane Demri and Ranko Lazić. Ltl with the freeze quantifier and register automata. *ACM Trans. Comput. Logic*, 10(3):16:1–16:30, 2009.

7 Diego Figueira. Satisfiability of downward XPath with data equality tests. In *PODS*, pages 197–206, 2009.

8 Diego Figueira. Forward-XPath and extended register automata on data-trees. In *ICDT*, pages 231–241, 2010.

9 Antonio Cano Gómez and Gloria Inés Alvarez. Learning commutative regular languages. In *ICGI*, pages 71–83, 2008.

10 Michael Kaminski and Nissim Francez. Finite-memory automata. *Theor. Comput. Sci.*, 134(2):329–363, November 1994.

11 Ahmet Kara, Thomas Schwentick, and Tony Tan. Feasible automata for two-variable logic with successor on data words. In *LATA*, pages 351–362, 2012. A long version can be found at http://arxiv.org/abs/1110.1221.

12 Amaldev Manuel and Ramaswamy Ramanujam. Class counting automata on datawords. *Int. J. Found. Comput. Sci.*, 22(4):863–882, 2011.

13 Andrew Martinez. Efficient computation of regular expressions from unary NFAs. In *DCFS*, pages 174–187, 2002.

14 Frank Neven, Thomas Schwentick, and Victor Vianu. Finite state machines for strings over infinite alphabets. *ACM Trans. Comput. Logic*, 5(3):403–435, 2004.

15 Jean Eric Pin. *Varieties of formal languages*. Plenum Publishers, 1986.

16 Luc Segoufin. Automata and logics for words and trees over an infinite alphabet. In *CSL, LNCS 4207*, pages 41–57, 2006.

17 H. Seidl, Th. Schwentick, A. Muscholl, and P. Habermehl. Counting in trees for free. In *ICALP*, pages 1136–1149, 2004.

18 Anthony Widjaja To. Unary finite automata vs. arithmetic progressions. *Inf. Process. Lett.*, 109(17):1010–1014, 2009.

19 Zhilin Wu. A decidable extension of data automata. In *GandALF*, pages 116–130, 2011.

A Proofs in Section 2

Proposition 2 ([15]). *Let L be a regular language over the alphabet $\Sigma = \{\sigma_1, \dots, \sigma_k\}$. Then L is commutative iff L is defined by a QFSP formula $\varphi(x_{\sigma_1}, \dots, x_{\sigma_k})$.*

Proof. "Only if" part:

Let's first consider the special case that the alphabet $\Sigma = \{\sigma\}$. Suppose L is defined by a finite automaton $\mathscr{A}$ with n states. It is well-known that a Chrobak normal form for $\mathcal{L}(\mathscr{A})$, i.e. a union of $O(n^2)$ arithmetic progressions $\sigma^r + \sigma^{p\mathbb{N}}$ (where $r = O(n^2), p \leq n$), can be constructed from $\mathscr{A}$ in polynomial time ([4, 13, 18]). For every arithmetic progression $\sigma^r + \sigma^{p\mathbb{N}}$, if $p = 0$, then the arithmetic progression is defined by the QFSP formula $x_\sigma = r$; if $p = 1$, then it is defined by the QFSP formula $x_\sigma \geq r$; otherwise, $p \geq 2$, then the arithmetic progression is defined by the QFSP formula $x_\sigma \geq r \wedge x_\sigma \equiv (r \bmod p) \bmod p$. Let φ be the disjunction of all these QFSP formulas corresponding to the arithmetic progressions, then L is defined by φ.

If the alphabet is not unary, let $\Sigma = \{\sigma_1, \dots, \sigma_k\}$ for some $k \geq 2$. Then for any word $v \in \Sigma^*$, $v \in L$ iff $\sigma_1^{\#_v(\sigma_1)} \dots \sigma_k^{\#_v(\sigma_k)} \in L$.

Suppose $\mathscr{A} = (Q, \Sigma, \delta, q_0, F)$ is a finite automaton with commutative transition relation δ such that $\mathcal{L}(\mathscr{A}) = L$.

Then $\sigma_1^{\#_v(\sigma_1)} \dots \sigma_k^{\#_v(\sigma_k)} \in L$ iff there exist $q_1, \dots, q_k$ such that $q_k \in F$ and $\sigma_i^{\#_v(\sigma_i)} \in \mathcal{L}(\mathscr{A}(q_{i-1}, q_i))$ for each $i : 1 \leq i \leq k$. From the above discussion for the unary alphabets, for each $i : 1 \leq i \leq k$, a QFSP formula $\varphi_{q_{i-1},q_i}(x_{\sigma_i})$ can be constructed from $\mathscr{A}$ to define the unary language $\left\{\sigma_i^j \in \mathcal{L}(\mathscr{A}(q_{i-1}, q_i)) \middle| j \in \mathbb{N}\right\}$.

Therefore, L is defined by the QFSP formula $\psi = \bigvee\limits_{\overline{q} \in Q^{k-1} \times F} \bigwedge\limits_{1 \leq i \leq k} \varphi_{q_{i-1},q_i}(x_{\sigma_i})$.

"If" part:

Let φ be a QFSP formula. we want to show that for any QFSP formula φ, $\mathcal{L}(\varphi)$ is a commutative regular language.

At first, it is not hard to prove that for each atomic QFSP formula φ, $\mathcal{L}(\varphi)$ is a commutative regular language. For instance, for the atomic formula $x_\sigma \equiv r \mod p$, there is a finite automaton which remembers the residual modulo p of the number of occurrences of the letter γ when reading a word from left to right, and accepts only when the residual is r.

Since QFSP formulas are Boolean combinations of atomic QFSP formulas and commutative regular languages are closed under all Boolean operations, it follows that the languages defined by QFSP formulas are commutative regular languages. ◀

Proposition 3. *Let $\varphi(x_1, \dots, x_k)$ be a QFSP formula. Then there exists an exponential-time algorithm to transform φ into a QFSP formula $\bigvee\limits_{i:1 \leq i \leq m} \varphi_i$ of size $2^{O(k|\varphi|)}$ such that there is $p_0 : 2 \leq p_0 \leq 2^{|\varphi|}$ satisfying that*

- *each φ_i is of the form $\bigwedge\limits_{1 \leq j \leq k} \varphi_{i,j}$;*
- *for each $j : 1 \leq j \leq k$, $\varphi_{i,j}$ is equal to $x_j = c_{i,j}$ or $x_j \geq p_0 \wedge x_j \equiv r_{i,j} \bmod p_0$ for $c_{i,j}, r_{ij} : 0 \leq c_{i,j}, r_{i,j} < p_0$;*
- *in addition, those φ_i's are mutually exclusive.*

Proof. Let $\varphi(x_1, \dots, x_k)$ be a QFSP formula.

Let p_1 be the least common multiplier of all those p occurring in the atomic formulas $x_{i_1} + \dots + x_{i_l} \equiv r \bmod p$ of φ. If there are no formulas of the form $x_{i_1} + \dots + x_{i_l} \equiv r \bmod p$ in φ, then let $p_1 = 2$. Let p_0 be the least multiplier of p_1 which is strictly greater than all the constants c occurring in the atomic formulas $x_{i_1} + \cdots + x_{i_l} \geq c$, or $x_{i_1} + \cdots + x_{i_l} \leq c$, or $x_{i_1} + \cdots + x_{i_l} = c$ of φ. It is easy to see that $p_0 \leq 2^{|\varphi|}$.

The formula $\varphi(x_1, \ldots, x_k)$ can be rewritten into the required form $\bigvee\limits_{i:1 \leq i \leq m} \varphi_i$ by the following procedure.

1. Rewrite φ into a formula with negation symbols only occurring before the atomic formulas.
2. Remove the negation symbols.
   - Replace $\neg(x_{i_1} + \cdots + x_{i_l} \leq c)$ by $x_{i_1} + \cdots + x_{i_l} \geq c + 1$.
   - If $c > 0$, then replace $\neg(x_{i_1} + \cdots + x_{i_l} \geq c)$ by $x_{i_1} + \cdots + x_{i_l} \leq c - 1$, otherwise, replace $\neg(x_{i_1} + \cdots + x_{i_l} \geq c)$ by $false$.
   - If $c > 0$, then replace $\neg(x_{i_1} + \cdots + x_{i_l} = c)$ by $x_{i_1} + \cdots + x_{i_l} \leq c - 1 \vee x_{i_1} + \cdots + x_{i_l} \geq c + 1$, otherwise, replace $\neg(x_{i_1} + \cdots + x_{i_l} = c)$ by $x_{i_1} + \cdots + x_{i_l} \geq 1$.
   - Replace $\neg(x_{i_1} + \cdots + x_{i_l} \equiv r \bmod p)$ by $\bigvee\limits_{0 \leq s < p, s \neq r} x_{i_1} + \cdots + x_{i_l} \equiv s \bmod p$.
3. Replace every atomic formula $x_{i_1} + \cdots + x_{i_l} \geq c$, $x_{i_1} + \cdots + x_{i_l} \leq c$, $x_{i_1} + \cdots + x_{i_l} = c$, and $x_{i_1} + \cdots + x_{i_l} \equiv r \bmod p$ by positive Boolean combinations of atomic formulas of the form $x_i \leq c$, $x_i \geq c$, $x_i = c$, and $x_i \equiv r \bmod p$.
   Let's take the formula $x + y \geq 3$ and $x + y \equiv 1 \bmod 3$ as examples to explain this step.

   The formula $x + y \geq 3$ is equivalent to
   $$x \geq 3 \vee y \geq 3 \vee (x = 2 \wedge y = 2) \vee (x = 1 \wedge y = 2) \vee (x = 2 \wedge y = 1).$$
   While the formula $x + y \equiv 1 \bmod 3$ is equivalent to
   $$(x \equiv 1 \bmod 3 \wedge y \equiv 0 \bmod 3) \vee (x \equiv 0 \bmod 3 \wedge y \equiv 1 \bmod 3) \vee (x \equiv 2 \bmod 3 \wedge y \equiv 2 \bmod 3).$$
4. Replace $x_i \leq c$ by $\bigvee\limits_{c' \leq c} x_i = c'$.
   Replace $x_i \geq c$ by $\left(\bigvee\limits_{c \leq c' < p_0} x_i = c'\right) \vee \bigvee\limits_{0 \leq r < p_0} (x_i \geq p_0 \wedge x_i \equiv r \bmod p_0)$.
   Replace $x_i \equiv r \bmod p$ by $\bigvee\limits_{j=0}^{t-1} x_i = jp + r \vee \bigvee\limits_{j=0}^{t-1} (x_i \geq p_0 \wedge x_i \equiv jp + r \bmod p_0)$, where $t = p_0/p$.
5. Rewrite the formula into a disjunctive normal form $\bigvee_i \psi_i$ and only keep those satisfiable disjuncts in $\bigvee_i \psi_i$.
   For each ψ_i and $j : 1 \leq j \leq k$, let $\psi_{i,j}$ be the conjunction of the atomic formulas involving x_j in ψ_i. If x_j does not appear in ψ_i, let $\psi_{i,j} := true$.
   It is not hard to see that each $\psi_{i,j}$ is of the form $true$, $x_j = c$, or $x_j \geq p_0 \wedge x_j \equiv r \bmod p_0$, where $0 \leq c, r < p_0$.
6. Make the disjuncts in the disjunctive normal form exclusive.
   For each formula $\psi_{i,j}$, if $\psi_{i,j} = true$, then replace $\psi_{i,j}$ by
   $$\left(\bigvee_{0 \leq c < p_0} x_j = c\right) \vee \bigvee_{0 \leq r < p_0} (x_j \geq p_0 \wedge x_j \equiv r \bmod p_0).$$
   Finally rewrite the formula into the disjunctive normal form again.

Complexity analysis.

By a simple calculation, we know that after the 4th step, each literal (i.e. atomic formula or negated atomic formula) in the original formula φ is replaced by a positive Boolean combination of at most $2^{O(k|\varphi|)}$ atomic formulas, so the size of the formula after the 4th step is $|\varphi| 2^{O(k|\varphi|)} = 2^{O(k|\varphi|)}$.
In the 5th step, the number of distinct disjuncts in the disjunctive normal form is at most $2^k(2p_0)^k = 2^{O(k|\varphi|)}$. The running time of the 5th step can be kept in $2^{O(k|\varphi|)}$ if some proper combinations and deletions of the disjuncts are applied during the rewriting process.
The complexity analysis of the 6th step is similar to the 5th step.
Therefore, a formula $\bigvee\limits_{i:1 \leq i \leq m} \varphi_i$ of the required form with $m = 2^{O(k|\varphi|)}$ can be obtained from φ in exponential time. Since for each i, $\varphi_i = \bigwedge\limits_{j:1 \leq j \leq k} \varphi_{i,j}$, and the size of each $\varphi_{i,j}$ is $O(\log k + \log p_0)$, it follows that the size of $\bigvee\limits_{i:1 \leq i \leq m} \varphi_i$ is $2^{O(k|\varphi|)} O(k(\log k + \log p_0)) = 2^{O(k|\varphi|)}$.

◀

B Proofs in Section 3

Theorem 4. *WDA < CDA < DA.*

Proof. We present the details of the proof that the language "In each class of the data word, the letter a occurs an even number of times" is not expressible in WDA.

Let L denote the language "In each class of the data word, the letter a occurs an even number of times".

To the contrary, suppose that L is defined by a WDA $\mathscr{D} = (\mathscr{A}, \mathscr{C})$, where $\mathscr{A} = (Q, \{a\} \times \{\perp, \top\}, \Gamma, \delta, F)$ and $\mathscr{C}$ is a collection of key constraints, inclusion constraints and denial constraints.

Let $k = |Q||\Gamma| + 1$. In addition, if $|\Gamma|$ is odd, then let $n = |\Gamma|^k + 1$, otherwise let $n = |\Gamma|^k + 2$. Note that n is defined to be even.

Suppose $w_1 = u^n$, where $u = \binom{a}{1} \cdots \binom{a}{k}$. Then $\mathsf{Profile}(w_1) = (a, \perp)^{kn}$.

Since n is even, it follows that $w_1 \in L$. So w_1 is accepted by $\mathscr{D}$, namely, there is an accepting run of $\mathscr{A}$ over w_1, say,

$$(q_0, (a, \perp), \gamma_1, q_1)(q_1, (a, \perp), \gamma_2, q_2) \ldots (q_{kn-1}, (a, \perp), \gamma_{kn}, q_{kn}),$$

such that the data word

$$w_1' = \left(\binom{\gamma_1}{1} \cdots \binom{\gamma_k}{k}\right) \cdots \left(\binom{\gamma_{k(n-1)+1}}{1} \cdots \binom{\gamma_{kn}}{k}\right)$$

satisfies all the constraints in $\mathscr{C}$.

Because $n \geq |\Gamma|^k + 1$, it follows that there are $i, j : 0 \leq i < j < n$ such that $\gamma_{ik+1} \cdots \gamma_{(i+1)k} = \gamma_{jk+1} \cdots \gamma_{(j+1)k}$. Therefore, for each $d : 1 \leq d \leq k$, the letter γ_{ik+d} occurs at least twice in the class of w_1' corresponding to d.

Consider the sequence $(\gamma_{ik+1}, q_{ik+1}) \ldots (\gamma_{(i+1)k}, q_{(i+1)k})$.

Because $k = |Q||\Gamma| + 1$, there are $r, s : 1 \leq r < s \leq k$ such that $(\gamma_{ik+r}, q_{ik+r}) = (\gamma_{ik+s}, q_{ik+s})$.

We distinguish between the two cases $s \geq r + 2$ and $s = r + 1$.

Case $s \geq r+2$.

Let w_2 be the data word

$$u^{ik}\binom{a}{1}\cdots\binom{a}{r}\left(\binom{a}{r+1}\cdots\binom{a}{s}\right)^2\binom{a}{s+1}\cdots\binom{a}{k}u^{(n-i-1)k}.$$

It is easy to see that w_2 is locally different, so $\mathsf{Profile}(w_2)=(a,\bot)^{kn+s-r}$. Then

$$\begin{array}{c}(q_0,(a,\bot),\gamma_1,q_1)\dots(q_{ik+r-1},(a,\bot),\gamma_{ik+r},q_{ik+r})\\ ((q_{ik+r},(a,\bot),\gamma_{ik+r+1},q_{ik+r+1})\dots(q_{ik+s-1},(a,\bot),\gamma_{ik+s},q_{ik+s}))^2\\ (q_{ik+s},(a,\bot),\gamma_{ik+s+1},q_{ik+s+1})\dots(q_{kn-1},(a,\bot),\gamma_{kn},q_{kn})\end{array}$$

is an accepting run of $\mathscr{A}$ over w_2. Let

$$w_2' = \begin{array}{c}\left(\binom{\gamma_1}{1}\cdots\binom{\gamma_k}{k}\right)\cdots\binom{\gamma_{ik+1}}{1}\cdots\binom{\gamma_{ik+r}}{r}\left(\binom{\gamma_{ik+r+1}}{r+1}\cdots\binom{\gamma_{ik+s}}{s}\right)^2\\ \binom{\gamma_{ik+s+1}}{s+1}\cdots\left(\binom{\gamma_{k(n-1)+1}}{1}\cdots\binom{\gamma_{kn}}{k}\right)\end{array}.$$

In the following, we show that w_2' satisfies all the constraints in $\mathscr{C}$. Because w_1' satisfies all the constraints in $\mathscr{C}$, it is sufficient to show that for each constraint in $\mathscr{C}$, w_1' satisfies it iff w_2' satisfies it.

For each $\gamma \in \Gamma$, let $D_{w_1'}(\gamma)$ be the set of data values occurring in the γ-positions in w_1'. Similarly $D_{w_2'}(\gamma)$ is defined for w_2'. From the above construction for w_2', it is not hard to see that for each $\gamma \in \Gamma$, $D_{w_1'}(\gamma)=D_{w_2'}(\gamma)$. Therefore, w_1' satisfies the inclusion and denial constraints in $\mathscr{C}$ iff w_2' satisfies them.

It remains to show that for each key constraint in $\mathscr{C}$, w_1' satisfies it iff w_2' satisfies it. It is sufficient to show that for each data value $d: 1 \leq d \leq k$ and $\gamma \in \Gamma$, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) \leq 1$ iff $\#_{\mathsf{Proj}(w_2'|_Y)}(\gamma) \leq 1$, where X (resp. Y) is the class of w_1' (resp. w_2') corresponding to d.

There are two situations.

- Let $d: 1 \leq d \leq r$, or $s+1 \leq d \leq k$, X (resp. Y) be the class of w_1' (resp. w_2') corresponding to d.
  From the definition of w_1' and w_2', we know that for every $\gamma \in \Gamma$, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) = \#_{\mathsf{Proj}(w_2'|_Y)}(\gamma)$, thus, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) \leq 1$ iff $\#_{\mathsf{Proj}(w_2'|_Y)}(\gamma) \leq 1$.
- Let $d: r+1 \leq d \leq s$, X (resp. Y) be the class of w_1' (resp. w_2') corresponding to d.
  For every $\gamma \in \Gamma$ such that $\gamma \notin \{\gamma_{ik+r+1}, \dots, \gamma_{ik+s}\}$, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) = \#_{\mathsf{Proj}(w_2'|_Y)}(\gamma)$.
  For every $\gamma \in \{\gamma_{ik+r+1}, \dots, \gamma_{ik+s}\}$, $\#_{\mathsf{Proj}(w_2'|_Y)}(\gamma) = \#_{\mathsf{Proj}(w_1'|_X)}(\gamma)+1$. Because we have $\gamma_{ik+1}\cdots\gamma_{i(k+1)} = \gamma_{jk+1}\cdots\gamma_{j(k+1)}$, it follows that for every data value $d': 1 \leq d' \leq k$ and every $\gamma \in \{\gamma_{ik+1}, \dots, \gamma_{i(k+1)}\}$, $\#_{\mathsf{Proj}(w_1'|_Z)}(\gamma) \geq 2$, where Z is the class of w_1' corresponding to d'. From this, we deduce that for every $\gamma \in \{\gamma_{ik+r+1}, \dots, \gamma_{ik+s}\}$, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) \geq 2$ and $\#_{\mathsf{Proj}(w_2'|_Y)}(\gamma) \geq 3$.
  Therefore, we conclude that for every $\gamma \in \Gamma$, $\#_{\mathsf{Proj}(w_1'|_X)}(\gamma) \leq 1$ iff $\#_{\mathsf{Proj}(w_2'|_Y)}(\gamma) \leq 1$.

Consequently, w_2' satisfies all the constraints in $\mathscr{C}$. It follows that w_2 is accepted by $(\mathscr{A},\mathscr{C})$, $w_2 \in L$.

On the other hand, for every data value $d: r+1 \leq d \leq s$, we have $\#_{\mathsf{Proj}(w_2|_Y)}(a) = \#_{\mathsf{Proj}(w_1|_X)}(a)+1$, where X (resp. Y) is the class of w_1 (resp. w_2) corresponding to d. From the fact that $w_1 \in L$, it follows that $w_2 \notin L$, a contradiction.

Case $s = r+1$.

Let w_2 be the data word

$$u^{ik}\binom{a}{1}\cdots\binom{a}{r}\binom{a}{r+1}\binom{a}{r}\binom{a}{r+2}\cdots\binom{a}{k}u^{(n-i-1)k}.$$

Since w_2 is locally different, $\mathsf{Profile}(w_2) = (a,\bot)^{kn+1}$. Because $(\gamma_{ik+r}, q_{ik+r}) = (\gamma_{ik+r+1}, q_{ik+r+1})$, it follows that

$$\begin{array}{c}(q_0,(a,\bot),\gamma_1,q_1)\dots(q_{ik+r-1},(a,\bot),\gamma_{ik+r},q_{ik+r})\\ (q_{ik+r},(a,\bot),\gamma_{ik+r+1},q_{ik+r+1})(q_{ik+r+1},(a,\bot),\gamma_{ik+r},q_{ik+r+1})\\ (q_{ik+r+1},(a,\bot),\gamma_{ik+r+2},q_{ik+r+2})\dots(q_{kn-1},(a,\bot),\gamma_{kn},q_{kn})\end{array}$$

is an accepting run of $\mathscr{A}$ over w_2. Let

$$w_2' = \begin{array}{c}\left(\binom{\gamma_1}{1}\cdots\binom{\gamma_k}{k}\right)\cdots\binom{\gamma_{ik+1}}{1}\cdots\binom{\gamma_{ik+r}}{r}\binom{\gamma_{ik+r+1}}{r+1}\binom{\gamma_{ik+r}}{r}\\ \binom{\gamma_{ik+r+2}}{r+2}\cdots\left(\binom{\gamma_{k(n-1)+1}}{1}\cdots\binom{\gamma_{kn}}{k}\right)\end{array}.$$

Similar to the argument for the case $s \geq r+2$, we can also show that w_2' satisfies all the constraints in $\mathscr{C}$, therefore, w_2 is accepted by $(\mathscr{A},\mathscr{C})$, so $w_2 \in L$.

On the other hand, for the class X corresponding to the data value r, there is one more occurrence of the letter a in the class X. Since $w_1 \in L$, it follows that $w_2 \notin L$, a contradiction as well. ◀

Theorem 5. *CDAs are closed under union and intersection, but not closed under complementation.*

Proof.

Union.

Let $\mathscr{D}_1 = (\mathscr{A}_1, \varphi_1)$ and $\mathscr{D}_2 = (\mathscr{A}_2, \varphi_2)$ be two CDAs.

Suppose for each $i = 1, 2$, $\mathscr{A}_i = (Q_i, \Sigma \times \{\bot,\top\}, \Gamma_i, \delta_i, q_{0,i}, F_i)$ and φ_i is a QFSP formula over the variable set V_{Γ_i}. Without loss of generality, we assume that $Q_1 \cap Q_2 = \emptyset$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$. If this is not the case, then it is easy to construct two CDAs $\mathscr{D}_1'$ and $\mathscr{D}_2'$ which are equivalent to $\mathscr{D}_1$ and $\mathscr{D}_2$ respectively and satisfy the above assumption.

Define $\mathscr{D} = (\mathscr{A}, \varphi)$ as follows.

- $\mathscr{A}$ nondeterministically chooses $\mathscr{A}_1$ or $\mathscr{A}_2$ to simulate. More formally, $\mathscr{A} := (Q, \Sigma \times \{\bot,\top\}, \Gamma_1 \cup \Gamma_2, \delta, q_0, F)$, where
  - $Q = Q_1 \cup Q_2 \cup \{q_0\}$,
  - δ is the union of δ_1 and δ_2 and the additional transitions from the state q_0 defined by the following rules,
    - if $(q_{0,1}, (\sigma, s), \gamma_1, q) \in \delta_1$, then $(q_0, (\sigma, s), \gamma_1, q) \in \delta$;
    - if $(q_{0,2}, (\sigma, s), \gamma_2, q) \in \delta_2$, then $(q_0, (\sigma, s), \gamma_2, q) \in \delta$.
  - $F = F_1 \cup F_2$.
- $\varphi := \left(\varphi_1 \wedge \sum\limits_{\gamma_1\in\Gamma_1} x_{\gamma_1} \geq 1\right) \vee \left(\varphi_2 \wedge \wedge \sum\limits_{\gamma_2\in\Gamma_2} x_{\gamma_2} \geq 1\right)$.

The two formulas $\sum\limits_{\gamma_1\in\Gamma_1} x_{\gamma_1} \geq 1$ and $\sum\limits_{\gamma_2\in\Gamma_2} x_{\gamma_2} \geq 1$ are used to avoid the possible interactions between $\mathscr{A}_i$ and φ_{3-i} in $(\mathscr{A}, \varphi)$ (where $i = 1, 2$).

Then $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\mathscr{D}_1) \cup \mathcal{L}(\mathscr{D}_2)$.

We would like to remark that the construction would be incorrect if we set $\varphi := \varphi_1 \vee \varphi_2$. For instance, let $\Sigma = \{a, b\}$, and $\mathscr{D}_1 = (\mathscr{A}_1, \varphi_1)$ and $\mathscr{D}_2 = (\mathscr{A}_2, \varphi_2)$ be two CDAs defined as follows.

- $\mathscr{A}_1$ is the transducer with the input alphabet Σ and it relabels each letter a (resp. b) by $(a, 1)$ (resp. $(b, 1)$), $\varphi_1 := x_{(a,1)} = 0$.
- $\mathscr{A}_2$ is the transducer with the input alphabet Σ and it relabels each letter a (resp. b) by $(a, 2)$ (resp. $(b, 2)$), $\varphi_2 := x_{(b,2)} = 0$.

Then $\mathcal{L}(\mathscr{D}_1)$ (resp. $\mathcal{L}(\mathscr{D}_2)$) is the set of data words which contains no letters a (resp. b). It follows that the union of $\mathcal{L}(\mathscr{D}_1)$ and $\mathcal{L}(\mathscr{D}_2)$ is the set of data words which contains either no letters a or no letters b.

If we let $\mathscr{A}$ be the CDA defined above and $\varphi := x_{(a,1)} = 0 \vee x_{(b,2)} = 0$, then $\mathcal{L}(\mathscr{A}, \varphi)$ would accept every data word over Σ: For each data word w over Σ, $\mathscr{A}$ chooses $\mathscr{A}_1$ to simulate and outputs $(a, 1)$ or $(b, 1)$ in each position, let w' be the resulting data word, then for each class X of w', $w'|_X \models x_{(b,2)} = 0$, therefore, $w'|_X \models \varphi$.

Intersection.

Let $\mathscr{D}_1 = (\mathscr{A}_1, \varphi_1)$ and $\mathscr{D}_2 = (\mathscr{A}_2, \varphi_2)$ be two CDAs.

Suppose for each $i = 1, 2$, $\mathscr{A}_i = (Q_i, \Sigma \times \{\bot, \top\}, \Gamma_i, \delta_i, q_{0,i}, F_i)$ and φ_i is a QFSP formula over the variable set V_{Γ_i}.

Define $\mathscr{D} = (\mathscr{A}, \varphi)$ as follows.

- $\mathscr{A} := (Q_1 \times Q_2, \Sigma \times \{\bot, \top\}, \Gamma_1 \times \Gamma_2, \delta, (q_{0,1}, q_{0,2}), F)$, where
  - δ is defined by the rule: $((q_1, q_2), (\sigma, s), (\gamma_1, \gamma_2), (q_1', q_2')) \in \delta$ iff $(q_1, (\sigma, s), \gamma_1, q_1') \in \delta_1$ and $(q_2, (\sigma, s), \gamma_2, q_2') \in \delta_2$.
  - $F = F_1 \times F_2$.
- $\varphi := \varphi_1' \wedge \varphi_2'$, where φ_1' is obtained from φ_1 by replacing each variable x_{γ_1} with $\sum\limits_{\gamma_2 \in \Gamma_2} x_{(\gamma_1, \gamma_2)}$ and φ_2' is obtained from φ_2 by replacing each variable x_{γ_2} with $\sum\limits_{\gamma_1 \in \Gamma_1} x_{(\gamma_1, \gamma_2)}$.

Then $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\mathscr{D}_1) \cap \mathcal{L}(\mathscr{D}_2)$.

Complementation.

Define the language L over the alphabet $\{a, b\}$ as the set of data words w satisfying the following conditions,

- $\mathsf{Proj}(w) \in a^+b^+$,
- each data value in w occurs exactly twice, one in a a-position, and the other in a b-position,
- the sequence of data values in a-positions is the same as the sequence of the data values in b-positions.

For instance, the data word $\binom{a}{1}\binom{a}{2}\binom{b}{1}\binom{b}{2}$ belongs to L, while $\binom{a}{1}\binom{a}{2}\binom{b}{2}\binom{b}{1}$ does not belong to L.

From [3], we know that L is not expressible in DA, thus not expressible in CDA as well.

In the following, we show that $\overline{L}$, the complement of L, can be recognized by a CDA. From this, we conclude that CDAs are not closed under complementation.

Let w be a data word such that $w \notin L$, then there are the following four situations,

1. $\mathsf{Proj}(w) \notin a^+b^+$,
2. there is a data value which occurs in a a-position but not in a b position, or occurs in a b-position but not in a a-position,
3. there are two a-positions or two b-positions with the same data value,

4. there are two a-positions, say $i, j : i < j$, and two b-positions, say $k, l : k < l$, such that the data value occurring in i is the same as that occurring in l and the data value occurring in j is the same as that occurring in k.

Then a CDA $\mathscr{D} = (\mathscr{A}, \varphi)$ to accept $\overline{L}$ is defined as the union of the CDAs $(\mathscr{A}_i, \varphi_i)$ (where $i = 1, 2, 3, 4$) defined in the following.

$(\mathscr{A}_1, \varphi_1)$:
- $\mathscr{A}_1$ verifies that $\mathsf{Proj}(w) \notin a^+b^+$.
- $\varphi_1 := true$.

$(\mathscr{A}_2, \varphi_2)$:
- $\mathscr{A}_2$ verifies that $\mathsf{Proj}(w) \in a^+b^+$, moreover, it either guesses an a-position and relabels it by $(a, \bigtriangledown)$, or guesses a b-position and relabels it by $(b, \bigtriangledown)$
- φ_2 is defined as follows,

$$\varphi_2 := (x_{(a,\bigtriangledown)} = 1 \wedge x_b = 0) \ \vee \ (x_{(b,\bigtriangledown)} = 1 \wedge x_a = 0) \ \vee (x_{(a,\bigtriangledown)} = 0 \wedge x_{(b,\bigtriangledown)} = 0).$$

$(\mathscr{A}_3, \varphi_3)$:
- $\mathscr{A}_3$ verifies that $\mathsf{Proj}(w) \in a^+b^+$, moreover, it either guesses two a-positions and relabels them by $\bigtriangleup$, or guesses two b-positions and relabels them by $\bigtriangleup$.
- $\varphi_3 := x_{\bigtriangleup} = 2 \ \vee \ x_{\bigtriangleup} = 0$.

$(\mathscr{A}_4, \varphi_4)$:
- $\mathscr{A}_4$ verifies that $\mathsf{Proj}(w) \in a^+b^+$, moreover, it guesses two a-positions $i, j : i < j$, relabels i by $\bigcirc$ and j by $\bigstar$, in addition, it guesses two b-positions $k, l : k < l$, relabels k by $\bigstar$ and l by $\bigcirc$.
- φ_4 is defined as follows,

$$\varphi_4 := (x_{\bigcirc} = 2 \wedge x_{\bigstar} = 0) \ \vee \ (x_{\bigstar} = 2 \wedge x_{\bigcirc} = 0) \vee (x_{\bigcirc} = 0 \wedge x_{\bigstar} = 0).$$

Since $w \in \overline{L}$ iff there is $i : 1 \leq i \leq 4$ such that w is accepted by $(\mathscr{A}_i, \varphi_i)$, it follows that $\mathcal{L}(\mathscr{A}, \varphi) = \overline{L}$. ◀

Theorem 6. *$EMSO^2_{\#}(+1, \sim, \Sigma)$ and CDA are expressively equivalent.*
- *Given a $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula $\exists R_1 \dots R_l(\varphi \wedge \forall x\psi)$, a CDA $\mathscr{D} = (\mathscr{A}, \varphi')$ of doubly exponential size can be constructed such that $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\exists R_1 \dots R_l(\varphi \wedge \forall x\psi))$. In addition, the size of the output alphabet of $\mathscr{A}$ is at most exponential over the size of $\exists R_1 \dots R_l(\varphi \wedge \forall x\psi)$.*
- *Given a CDA $\mathscr{D} = (\mathscr{A}, \varphi)$, a $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula φ' of polynomial size can be constructed such that $\mathcal{L}(\mathscr{D}) = \mathcal{L}(\varphi')$.*

Proof. *From $EMSO^2_{\#}(+1, \sim, \Sigma)$ to CDA.*

Let $\exists R_1 \dots R_l(\varphi \wedge \forall x\psi)$ be a $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula.

According to the results in [11], from the $EMSO^2(+1, \sim, \Sigma)$ formula $\exists R_1 \dots R_l\varphi$, an equivalent WDA $(\mathscr{A}, \mathscr{C})$ of doubly exponential size can be constructed. In particular, the output alphabet of $\mathscr{A}$, denoted by Γ, is $\Sigma \times 2^{\{R_1, \dots, R_l, R_{l+1}, \dots, R_m\}}$, where $R_{l+1}, \dots, R_m$ are some additional monadic predicates (besides $R_1, \dots, R_l$); in addition, the size of Γ and the number of constraints in $\mathscr{C}$ are both at most exponential over the size of φ.

Let φ' be the QFSP formula constructed from $\mathscr{C}$ and ψ by the following procedure.

1. Construct a QFSP formula $\eta_{\mathscr{C}}$ from $\mathscr{C}$ as follows: $\eta_{\mathscr{C}} := \bigwedge\limits_{C \in \mathscr{C}} \eta_C$, where η_C is defined by the following rules,
   - if C is a key constraint $\mathsf{Key}(\gamma)$, then $\eta_C := x_\gamma \leq 1$,

- if C is an inclusion constraint $D(\gamma) \subseteq \bigcup\limits_{\gamma' \in R} D(\gamma')$, then $\eta_C := x_\gamma \geq 1 \rightarrow \sum\limits_{\gamma' \in R} x_{\gamma'} \geq 1$,
- if C is a denial constraint $D(\gamma) \cup D(\gamma') = \emptyset$, then $\eta_C := x_\gamma \geq 1 \rightarrow x_{\gamma'} = 0$.

2. Construct a QFSP formula ψ' from ψ as follows.
   - Replace each formula of the form $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \geq c$ in ψ (where $\Delta \subseteq \Sigma \times 2^{\{R_1,\dots,R_l\}}$) by $\sum\limits_{\gamma : (\gamma|_{\Sigma \times 2^{\{R_1,\dots,R_l\}}}) \in \Delta} x_\gamma \geq c$, where $\gamma|_{\Sigma \times 2^{\{R_1,\dots,R_l\}}}$ is the projection of γ to $\Sigma \times 2^{\{R_1,\dots,R_l\}}$ for every $\gamma \in \Gamma$.
   - Apply similar replacements for every formulas of the form $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \leq c$, $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) = c$, and $\sum\limits_{\tau \in \Delta} \#_{x \sim y \wedge \tau(y)}(y) \equiv r \bmod p$.
3. Let $\varphi' := \eta_{\mathscr{C}} \wedge \psi'$.

Then the CDA $(\mathscr{A}, \varphi')$ is equivalent to $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$.

From the construction, we know that the size of ψ' is at most exponential over the size of ψ. Because both the size of Γ and the number of constraints in $\mathscr{C}$ are at most exponential over the size of $\exists R_1 \dots R_l \varphi$, it follows that the size of $\eta_{\mathscr{C}}$ is at most exponential over the size of $\exists R_1 \dots R_l \varphi$. Therefore, the size of φ' is at most exponential over the size of $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$.

Since the size of $\mathscr{A}$ is also at most doubly exponential over the size of $\exists R_1 \dots R_l \varphi$, it follows that the size of the CDA $(\mathscr{A}, \varphi')$ is at most doubly exponential over the size of $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$. In addition, the size of the output alphabet of $\mathscr{A}$, namely Γ, is at most exponential over the size of $\exists R_1 \dots R_l(\varphi \wedge \forall x \psi)$.

From CDA to $EMSO^2_{\#}(+1, \sim, \Sigma)$.

Let $\mathscr{D} = (\mathscr{A}, \varphi)$ be a CDA such that $\mathscr{A} = (Q, \Sigma \times \{\bot, \top\}, \Gamma, \delta, q_0, F)$ and φ be a QFSP formula over the variable set V_Γ. Suppose $Q = \{q_0, q_1, \dots, q_n\}$ and $\Gamma = \{\gamma_1, \dots, \gamma_l\}$. In addition, it can be assumed that the initial state q_0 only occurs in the beginning of any run of $\mathscr{A}$.

As proved in [11], from a WDA $(\mathscr{A}, \mathscr{C})$, an equivalent $EMSO^2(+1, \sim, \Sigma)$ formula η of the following form can be constructed,

$$\eta := \exists R_{q_1} \dots R_{q_n} R_{\gamma_1} \dots R_{\gamma_l} (\varphi_{part} \wedge \varphi_{start} \wedge \varphi_{trans} \wedge \varphi_{accept} \wedge \varphi_{constr}),$$

where

- the formula φ_{part} describes the fact that both $(R_{q_i})_{1 \leq i \leq n}$ and $(R_{\gamma_j})_{1 \leq j \leq l}$ form a partition of all the positions in a run,
- the formula φ_{start}, φ_{trans} and φ_{accept} describe respectively the state in the first position, the transition relation and the acceptance condition of the transducer $\mathscr{A}$,
- and the formula φ_{constr} describes all the constraints in $\mathscr{C}$.

Let ψ be the formula obtained from the QFSP formula φ by replacing the formulas of the form $x_{\gamma_{j_1}} + \dots + x_{\gamma_{j_k}} \geq c$ with

$$\left(\sum_{s=1}^{n} \sum_{t=1}^{k} \#_{x \sim y \wedge R_{q_s}(y) \wedge R_{\gamma_{j_t}}(y)}(y) \geq c \right),$$

and applying similar replacements for the formulas of the form $x_{\gamma_{j_1}} + \dots + x_{\gamma_{j_k}} \leq c$, $x_{\gamma_{j_1}} + \dots + x_{\gamma_{j_k}} = c$, and $x_{\gamma_{j_1}} + \dots + x_{\gamma_{j_k}} \equiv r \bmod p$.

Then the desired $EMSO^2_{\#}(+1, \sim, \Sigma)$ formula φ' is obtained from η by replacing φ_{constr} with $\forall x \psi$,

$$\varphi' := \exists R_{q_1} \dots R_{q_n} R_{\gamma_1} \dots R_{\gamma_l} (\varphi_{part} \wedge \varphi_{start} \wedge \varphi_{trans} \wedge \varphi_{accept} \wedge \forall x \psi).$$

It is a routine to check that $\mathcal{L}(\varphi') = \mathcal{L}(\mathscr{D})$.

Since the size of each of the formulas $\varphi_{part}, \varphi_{start}, \varphi_{trans}, \varphi_{accept}$ is polynomial over the size of $\mathscr{A}$, and the size of ψ is polynomial over the size of $\mathscr{A}$ and the size of φ, it follows that the size of φ' is polynomial over the size of $\mathscr{D}$. ◀

C Proofs in Section 4

C.1 Reduction to NONEMPTINESS-LOCALLY-DIFFERENT

Suppose $\mathscr{A} = (Q, \Gamma \times \{\bot, \top\}, \delta, q_0, F)$ and φ is a QFSP formula over the variable set V_Γ.

Let $p_0 \geq 2$ be the normalization number of φ and $\psi = \bigvee\limits_{1 \leq i \leq m} \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$ be the normal form of φ.

Let $v = \gamma_1 \dots \gamma_n \in \Gamma^*$. A *summary* of an interval $[i, j]$ of v, denoted by $\mathcal{S}([i, j])$, is a function $\theta : \Gamma \to \{\geq, <\} \times \{0, \dots, p_0 - 1\}$ such that

- $\theta(\gamma) = (\geq, r)$ iff there are at least p_0 γ-positions in the interval $[i, j]$ and the number of γ-positions in the interval is equal to r modulo p_0.
- $\theta(\gamma) = (<, c)$ iff there are exactly c γ-positions in the interval (where $c < p_0$).

Let $w = \binom{\gamma_1}{d_1}\binom{\gamma_2}{d_2} \dots \binom{\gamma_n}{d_n}$ be a data word over Γ. A *zone* of w is a maximal interval $[i, j]$ with the same data value, i.e. $d_i = d_{i+1} = \dots = d_j$, $d_{i-1} = d_i$ (if $i > 1$) and $d_j \neq d_{j+1}$ (if $j < n$). Let $\theta : \Gamma \to \{\geq, <\} \times \{0, \dots, p_0 - 1\}$, then a zone is called a *θ-zone* if the summary of the zone is θ. Let $[1, k_1], [k_1 + 1, k_2], \dots, [k_{l-1}, n]$ be the collection of all zones in w. Then the *zonal* word of w, denoted by $\mathsf{Zonal}(w)$, is a data word over the alphabet $\Gamma \cup (\{\geq, <\} \times \{0, \dots, p_0 - 1\})^\Gamma$ defined as follows,

$$\mathsf{Zonal}(w) = \gamma_1 \dots \gamma_{k_1} \binom{\theta_1}{d_{k_1}} \gamma_{k_1+1} \dots \gamma_{k_2} \binom{\theta_2}{d_{k_2}} \dots \gamma_{k_{l-1}+1} \dots \gamma_n \binom{\theta_l}{d_n},$$

where for each $i : 0 \leq i < l$, $\theta_i = \mathcal{S}([k_i + 1, k_{i+1}])$ ($k_0 = 0, k_l = n$ by convention).

Note that the sequence of data values $d_{k_1} d_{k_2} \dots d_n$ in $\mathsf{Zonal}(w)$ is locally different.

In general, we call words of the form

$$\gamma_1 \dots \gamma_{k_1} \binom{\theta_1}{d_1} \gamma_{k_1+1} \dots \gamma_{k_2} \binom{\theta_2}{d_2} \dots \gamma_{k_{l-1}+1} \dots \gamma_n \binom{\theta_l}{d_l},$$

with the sequence of data values $d_1 d_2 \dots d_l$ not necessarily locally different, as *pseudo-zonal* words. A pseudo-zonal word is called locally different if the sequence of data values occurring in it is locally different.

Let $\Theta = (\{\geq, <\} \times \{0, \dots, p_0 - 1\})^\Gamma$ and $\Gamma' = \Gamma \cup \Theta$.

In the following, we will construct a finite automaton $\mathscr{A}'$ over the alphabet Γ' and a QFSP formula φ' over the variable set $V_{\Gamma'}$ in the normal form such that there is a data word w accepted by $(\mathscr{A}, \varphi)$ iff there is a locally different pseudo-zonal word w' accepted by $(\mathscr{A}', \varphi')$.

- The finite automaton $\mathscr{A}'$ over the alphabet Γ'.
  Over a word $w = \gamma_1 \dots \gamma_{k_1}\ \theta_1\ \gamma_{k_1+1} \dots \gamma_{k_2}\ \theta_2\ \dots \gamma_{k_{l-1}+1} \dots \gamma_n\ \theta_l$ (where $\gamma_1, \dots, \gamma_n \in \Gamma$ and $\theta_1, \dots, \theta_l \in \Theta$), which corresponds to the projection of a pseudo-zonal word

  $$w = \gamma_1 \dots \gamma_{k_1} \begin{pmatrix}\theta_1\\ d_1\end{pmatrix} \gamma_{k_1+1} \dots \gamma_{k_2} \begin{pmatrix}\theta_2\\ d_2\end{pmatrix} \dots \gamma_{k_{l-1}+1} \dots \gamma_n \begin{pmatrix}\theta_l\\ d_l\end{pmatrix},$$

  the finite automaton $\mathscr{A}'$
  - simulates the run of $\mathscr{A}$ over the word

    $$(\gamma_1, \top) \dots (\gamma_{k_1}, \bot)(\gamma_{k_1+1}, \top) \dots (\gamma_{k_2}, \bot) \dots (\gamma_{k_{l-1}+1}, \top) \dots (\gamma_n, \bot),$$

  - checks whether $\mathcal{S}([k_i + 1, k_{i+1}]) = \theta_i$ for each $i : 0 \leq i < l$ (where $k_0 = 0, k_l = n$).

  We remark that $\mathscr{A}'$ runs on the projections of the pseudo-zonal words, instead of the profile words of them.
- The QFSP formula φ' over the variable set $V_{\Gamma'}$.
  The formula φ' is in fact a QFSP formula over the variable set $V_\Theta = \{x_\theta \mid \theta \in \Theta\}$. It is constructed from ψ (the normal form of φ) by the following procedure.
  1. Replace each atomic formula $x_\gamma = c$ by

     $$\left(\bigwedge_{\theta:\pi_1(\theta(\gamma))=\geq} x_\theta = 0 \right) \bigwedge \left(\sum_{\theta:\pi_1(\theta(\gamma))=<} \pi_2(\theta(\gamma)) x_\theta = c \right),$$

     where $\pi_1(\theta(\gamma))$ and $\pi_2(\theta(\gamma))$ are respectively the first and second component of $\theta(\gamma)$.
  2. Replace each formula $x_\gamma \geq p_0 \wedge x_\gamma \equiv r \bmod p_0$ by

     $$\left(\bigvee_{\theta:\pi_1(\theta(\gamma))=\geq} x_\theta \geq 1 \bigvee \sum_{\theta:\pi_1(\theta(\gamma))=<} \pi_2(\theta(\gamma)) x_\theta \geq p_0 \right) \bigwedge \sum_{\theta} \pi_2(\theta(\gamma)) x_\theta \equiv r \bmod p_0.$$

  3. Rewrite the formula into the normal form.

Consequently we have reduced the problem of NONEMPTINESS-PROFILE to the following problem:

Decide whether there is a locally different pseudo-zonal word w accepted by $(\mathscr{A}', \varphi')$.

Essentially this problem is the same as the problem of NONEMPTINESS-LOCALLY-DIFFERENT, except some minor technical differences. Therefore, once we know how to solve NONEMPTINESS-LOCALLY-DIFFERENT, we also know how to solve this problem (with the same complexity).

C.2 Complexity analysis

Algorithm for NONEMPTINESS-LOCALLY-DIFFERENT-MANY.

The first step of the algorithm is in polynomial time since the size of ψ is polynomial over the size of φ according to Lemma 10. From Theorem 1, it follows that the second step can be solved in NP. Therefore, the complexity of the whole algorithm is in NP.

Algorithm for NONEMPTINESS-LOCALLY-DIFFERENT.

Since $h_i \leq 2|\Gamma|p_0$ for every $i : 1 \leq i \leq m$, it follows that the size of $\Gamma \times D_J$ is at most

$$|\Gamma| \sum_{1 \leq i \leq m} \max(2p_0 + 1, 2h_i + 3) \leq m|\Gamma|(4|\Gamma|p_0 + 3).$$

For each $(\gamma, d) \in \Gamma \times D_J$, the automaton $\mathscr{A}'$ need remember the residual of the number of (γ, d)-positions modulo p_0 and whether it is greater than p_0.
Therefore, the size of $\mathscr{A}'$ is $O(|Q|(2p_0)^{|\Gamma \times D_J|}) = O(|Q|(2p_0)^{m|\Gamma|(4|\Gamma|p_0+3)})$.
Because $m \leq |\varphi|$ and $p_0 \leq 2^{|\varphi|}$, it follows that the size of $\mathscr{A}'$ is $O(|Q|)2^{O(|\varphi|^2|\Gamma|^2 2^{|\varphi|})}$ by simple calculations.
The lengths of the binary encodings of p_0 and h_i's are polynomial over $|\varphi|$. In addition, $|\psi|$ and $|J|$ are both polynomial over $|\varphi|$. It follows that the size of $\psi_{g,J}$ is polynomial over $|\varphi|$.
From Theorem 1, we know that the last step of the algorithm is in NP. It follows that the complexity of the whole algorithm is in 2-NEXPTIME.

Reduction of NONEMPTINESS-PROFILE to NONEMPTINESS-LOCALLY-DIFFERENT.

From Proposition 3, we know that the size of ψ (the normal form of φ) is $2^{O(|\Gamma||\varphi|)}$ and $p_0 \leq 2^{|\varphi|}$.
The size of $\mathscr{A}'$ is $O(|Q||\Theta|) = O(|Q|(2p_0)^{|\Gamma|}) \leq |Q|2^{O(|\Gamma||\varphi|)}$. In addition, the size of the alphabet $\Gamma' = \Gamma \cup \Theta$ is $2^{O(|\Gamma||\varphi|)}$.
After the first two steps of the procedure to construct φ' from ψ, the size of the formula becomes $O(|\psi||\Theta| \log p_0 \log|\Theta|) = 2^{O(|\Gamma||\varphi|)}$. The third step of the procedure transforms the formula into a normal form. From Proposition 3, it follows that the size of φ' is $2^{|\Theta|2^{O(|\Gamma||\varphi|)}} = 2^{(2p_0)^{|\Gamma|}2^{O(|\Gamma||\varphi|)}} = 2^{2^{O(|\Gamma||\varphi|)}}$. Note that the normalization number of φ' is still $p_0 \leq 2^{|\varphi|}$.

Algorithm for NONEMPTINESS-PROFILE.

Let $\mathscr{A} = (Q, \Gamma \times \{\bot, \top\}, \delta, q_0, F)$ be a finite automaton over the alphabet $\Gamma \times \{\bot, \top\}$ and φ be a QFSP formula over the variable set V_Γ.
The nonemptiness of $(\mathscr{A}, \varphi)$ is decided by the following procedure.

1. Construct from $(\mathscr{A}, \varphi)$ a finite automaton $\mathscr{A}' = (Q', \Gamma', \delta', q_0', F')$ and a QFSP formula $\varphi' = \bigvee_{1 \leq i \leq m} \varphi_i'$ over the variable set $V_{\Gamma'}$ in the normal form (c.f. Section C.1).
2. Solve the problem of NONEMPTINESS-LOCALLY-DIFFERENT with input $(\mathscr{A}', \varphi')$ (c.f. Section 4.2).
   a. Guess a set $J \subseteq \{1, \ldots, m\}$ and a set of data values D_J.
   b. Construct a finite automaton $\mathscr{A}'' = (Q'', \Gamma' \cup \Gamma' \times D_J, \delta'', q_0'', F'')$ and an existential Presburger formula $\psi_{g,J}$.
   c. Decide the nonemptiness of the Presburger automaton $(\mathscr{A}'', \psi_{g,J})$.

From the above analysis, we know that

- $|Q'|$, the size of $\mathscr{A}'$, is $|Q|2^{O(|\Gamma||\varphi|)}$,
- the size of Γ' is $2^{O(|\Gamma||\varphi|)}$,
- the size of φ' is $2^{2^{O(|\Gamma||\varphi|)}}$,
- p_0, the normalization number of φ', is at most $2^{|\varphi|}$,
- the size of $\mathscr{A}''$ is

$$O\left(|Q'|(2p_0)^{m|\Gamma'|(4|\Gamma'|p_0+3)}\right) = O\left(|Q'|(2p_0)^{|\varphi'||\Gamma'|(4|\Gamma'|p_0+3)}\right) = O(|Q|)2^{2^{2^{O(|\Gamma||\varphi|)}}},$$

- the size of $\psi_{g,J}$ is polynomial over $|\varphi'|$, thus $2^{2^{O(|\Gamma||\varphi|)}}$.

Since the nonemptiness of $(\mathscr{A}'', \psi_{g,J})$ can be decided in NP (Theorem 1), we conclude that the complexity of the whole algorithm is in 3-NEXPTIME.

D Proofs in Section 5

Theorem 15. *The nonemptiness of CBDA can be decided in 4-NEXPTIME.*

Proof. We illustrate a nondeterministic exponential-time reduction of nonemptiness of CBDA to that of CDA.

The main idea of the reduction is the same as the reduction of weak Büchi data automata (WBDA) to WDA in [11].

In the reduction of WBDA to WDA, from a given WBDA, a nondeterministic polynomial time algorithm constructs a WDA that accepts the *finite witnesses* for the WBDA. Similarly, in the following, from a given CBDA, we will present a nondeterministic exponential time algorithm to construct a CDA that accepts the finite witnesses for the CBDA. Nevertheless, the reduction becomes more involved in the following sense.

- There is an exponential increase in size of the CDA that accepts the finite witnesses for the CBDA, compared to that for WBDA, as a result of the modular constraints in the class conditions.
- It is necessary to differentiate the labels that occur infinitely many times from those that occur only finitely many times, because of the ω-constraints (i.e. constraints of the form $x_\gamma = \omega$) in the class conditions.

Let $(\mathscr{A}, \varphi)$ be a CBDA such that $\mathscr{A} = (Q, \Sigma \times \{\bot, \top\}, \Gamma, \delta, q_0, F)$ and φ is a ω-QFSP formula over V_Γ.

Because we are concerned with the nonemptiness problem, we can assume that $\mathscr{A}$ is just a Büchi automaton $(Q, \Gamma \times \{\bot, \top\}, \delta, q_0, F)$. Then the nonemptiness problem becomes into the following problem,

> "is there a data word w over the alphabet Γ such that $\mathsf{Profile}(w)$ is accepted by $\mathscr{A}$ and $w \models_c \varphi$?"

In addition, we assume that φ is in the normal form $\bigvee\limits_{1 \le i \le m} \bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$, and we will discuss later the general case that φ is not in the normal form. Let p_0 be the normalization number of φ.

We first introduce some notations.

An *adorned zone* of a data word w is a zone together with a state-pair (q, q') (where $q, q' \in Q$). For a zone z of w adorned with a state-pair (q, q'), the triple $(\mathsf{Proj}(z), q, q')$ is called the *adorned projection* of z in w, denoted by $\mathsf{a\text{-}Proj}_w(z)$.

A *singular witness* for $(\mathscr{A}, \varphi)$ is a data word uv over $\Gamma \times \{0, 1\}$ satisfying the following properties.

- $\mathsf{Proj}(u) \in (\Gamma \times \{0\})^*$ and $\mathsf{Proj}(v) \in (\Gamma \times \{1\})^*$.
- $uv \models_c \varphi'$, where φ' is constructed from φ by applying the following replacements.
  - If $\varphi_{i,\gamma} := x_\gamma = c_{i,\gamma}$, then replace $\varphi_{i,\gamma}$ by $\varphi'_{i,\gamma} := x_{(\gamma,0)} = c_{i,\gamma} \wedge x_{(\gamma,1)} = 0$. Intuitively, the formula $x_{(\gamma,1)} = 0$ means that the letter γ does not occur in v.
  - If $\varphi_{i,\gamma} := x_\gamma \ge p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$, then replace $\varphi_{i,\gamma}$ by $\varphi'_{i,\gamma} := x_{(\gamma,0)} \ge p_0 \wedge x_{(\gamma,0)} \equiv r_{i,\gamma} \bmod p_0 \wedge x_{(\gamma,1)} = 0$.
  - If $\varphi_{i,\gamma} := x_\gamma = \omega$, then replace $\varphi_{i,\gamma}$ by $\varphi'_{i,\gamma} := x_{(\gamma,1)} \ge 1$. Intuitively, the formula $x_{(\gamma,1)} \ge 1$ guarantees that the letter γ occurs in v.

- All the positions of v and the last zone of u have the same data value.
- There is a state $\hat{q} \in F$ and a (partial) run of $\mathscr{A}$ over $\mathsf{Profile}(u'v')_\top$, say $\rho = \rho_u \rho_v$, such that the state of ρ after reading $\mathsf{Profile}(u')$ and the state after reading $\mathsf{Profile}(u'v')_\top$ are both $\hat{q}$, where u' (resp. v') is the data word over Γ obtained from u (resp. v) by replacing each letter (γ, b) (where $b \in \{0,1\}$) with γ, and $\mathsf{Profile}(u'v')_\top$ denotes the profile of $u'v'$ with the last profile symbol replaced by $\top$.

Let $\Lambda = \Gamma \times \{0,1\} \times \{\mathsf{black}, \mathsf{white}, \mathsf{blue}\} \cup \Gamma \times \{1\} \times \{\mathsf{yellow}\} \times \{\mathsf{fin}, \mathsf{inf}\}$. For every $\mathsf{color} \in \{\mathsf{black}, \mathsf{white}, \mathsf{blue}\}$ and $b \in \{0,1\}$, let $\Lambda_{\mathsf{color},b} = \Gamma \times \{b\} \times \{\mathsf{color}\}$. In addition, for every $\mathsf{color} \in \{\mathsf{black}, \mathsf{white}, \mathsf{blue}\}$, let $\Lambda_{\mathsf{color}} = \Lambda_{\mathsf{color},0} \cup \Lambda_{\mathsf{color},1}$. Finally, let $\Lambda_{\mathsf{yellow}} = \Gamma \times \{1\} \times \{\mathsf{yellow}\} \times \{\mathsf{fin}, \mathsf{inf}\}$.

A *non-singular witness* for $(\mathscr{A}, \varphi)$ is a data word uv over the alphabet Λ satisfying the following conditions.

- $u \in (\Lambda_{\mathsf{black},0} \cup \Lambda_{\mathsf{white},0} \cup \Lambda_{\mathsf{blue},0})^*$,
  $v \in (\Lambda_{\mathsf{black},1} \cup \Lambda_{\mathsf{white},1} \cup \Lambda_{\mathsf{blue},1} \cup \Lambda_{\mathsf{yellow}})^*$.
- $uv \models_c \varphi'$, where φ' is obtained from φ by the following procedure.
  1. Construct the formula η which asserts that each class carries one color,
  $$\eta := \bigwedge_{\mathsf{color} \in \{\mathsf{black},\mathsf{white},\mathsf{yellow},\mathsf{blue}\}} \left(\begin{array}{c} \left(\bigvee_{\lambda \in \Lambda_{\mathsf{color}}} (x_\lambda \geq 1) \right) \to \\ \bigwedge_{\mathsf{color}' \neq \mathsf{color}} \bigwedge_{\lambda \in \Lambda_{\mathsf{color}'}} (x_\lambda = 0) \end{array} \right).$$
  2. Construct ψ from φ by applying the following replacements.
     - Replace $\varphi_{i,\gamma} := x_\gamma = c_{i,\gamma}$ by
     $$\begin{aligned} \varphi'_{i,\gamma} \quad := \bigvee_{\mathsf{color} \in \{\mathsf{black},\mathsf{white}\}} & (x_{(\gamma,0,\mathsf{color})} = c_{i,\gamma} \wedge x_{(\gamma,1,\mathsf{color})} = 0) \bigvee \\ & (x_{(\gamma,1,\mathsf{yellow},\mathsf{fin})} = c_{i,\gamma} \wedge x_{(\gamma,1,\mathsf{yellow},\mathsf{inf})} = 0). \end{aligned}$$
     - Replace $\varphi_{i,\gamma} := x_\gamma \geq p_0 \wedge x_\gamma \equiv r_{i,\gamma} \bmod p_0$ by
     $$\begin{aligned} \varphi'_{i,\gamma} \quad := \bigvee_{\mathsf{color} \in \{\mathsf{black},\mathsf{white}\}} & (x_{(\gamma,0,\mathsf{color})} \geq p_0 \wedge x_{(\gamma,0,\mathsf{color})} \equiv r_{i,\gamma} \bmod p_0 \wedge x_{(\gamma,1,\mathsf{color})} = 0) \bigvee \\ & (x_{(\gamma,1,\mathsf{yellow},\mathsf{fin})} \geq p_0 \wedge x_{(\gamma,1,\mathsf{yellow},\mathsf{fin})} \equiv r_{i,\gamma} \bmod p_0 \wedge x_{(\gamma,1,\mathsf{yellow},\mathsf{inf})} = 0). \end{aligned}$$
     - Replace $\varphi_{i,\gamma} := x_\gamma = \omega$ by
     $$\varphi'_{i,\gamma} := \left(\bigvee_{\mathsf{color} \in \{\mathsf{black},\mathsf{white}\}} x_{(\gamma,1,\mathsf{color})} \geq 1 \right) \bigvee x_{(\gamma,1,\mathsf{yellow},\mathsf{inf})} \geq 1.$$

     Note that the blue color is not included in the construction of ψ. Intuitively, this means that the blue classes are not required to satisfy the class condition φ.
  3. Let $\varphi' := \eta \wedge \psi$.
- All zones are of length at most $|Q|(2p_0)^{|\Gamma|}$.
- Every yellow zone z satisfy the condition that z does not contain the occurrences of "fin" and "inf" simultaneously, more formally, for any $\gamma, \gamma' \in \Gamma$, there do not exist two distinct positions in z labeled by $(\gamma, 1, \mathsf{yellow}, \mathsf{fin})$ and $(\gamma', 1, \mathsf{yellow}, \mathsf{inf})$ respectively. Therefore, each yellow zone z can be called a "fin"-zone or a "inf"-zone, depending on the occurrences of "fin" or "inf" in z.
- The last position of u is the last position of a zone in uv (This implies that the data value in the last position of u is different from that in the first position of v). The last zone of v is a non-black and non-white zone.

- There is a state $\hat{q} \in F$ and a (partial) run of $\mathscr{A}$ over $\mathsf{Profile}(u'v')$, say $\rho = \rho_u \rho_v$, such that the state of ρ after reading $\mathsf{Profile}(u')$ and the state after reading $\mathsf{Profile}(u'v')$ are both $\hat{q}$, where u' (resp. v') is the data word over Γ obtained from u (resp. v) by replacing each letter $(\gamma, \dots) \in \Lambda$ with γ. In the following, each zone z is adorned by the pair (q, q') where q is the state of ρ before reading z and q' is the state after reading z; in addition, for a zone z of $u'v'$ with the adornment (q, q'), $\mathsf{a\text{-}Proj}_{u'v'}(z)$ is used to denote $(\mathsf{Proj}(z), q, q')$.
- There are at most $|Q|^2$ yellow classes in uv, with each of them containing at most $\left((2p_0)^{|\Gamma|} + |Q|^2 (2p_0)^{2|\Gamma|}\right)$ "fin"-zones and at most $|\Gamma|$ "inf"-zones.
- For each blue zone z, there is a yellow "fin"-zone z' such that $\mathsf{a\text{-}Proj}_{u'v'}(z) = \mathsf{a\text{-}Proj}_{u'v'}(z')$.

The proof of the theorem is reduced to proving the following three claims.

Claim 1 If there is a witness for $(\mathscr{A}, \varphi)$ then $\mathcal{L}((\mathscr{A}, \varphi)) \neq \emptyset$.

Claim 2 If $\mathcal{L}((\mathscr{A}, \varphi)) \neq \emptyset$, then there is a witness for $(\mathscr{A}, \varphi)$.

Claim 3 There is a nondeterministic exponential time algorithm such the over the input $(\mathscr{A}, \varphi)$,

- in each run of the algorithm, a CDA $(\mathscr{B}, \psi)$ which accepts only witnesses for $(\mathscr{A}, \varphi)$ is constructed,
- for each witness uv of $(\mathscr{A}, \varphi)$, there is a run of the algorithm which constructs a CDA $(\mathscr{B}, \psi)$ that accepts uv.

The proofs of the three claims will be put in the next three subsections. Then in the last subsection, the complexity of the nondeterministic algorithm will be analysed and the discussion of the general case that φ is not in the normal form will be presented. ◀

D.1 Proof of Claim 1

We distinguish between singular and non-singular witnesses.

Let uv be a witness for $(\mathscr{A}, \varphi)$.

The situation that uv is a singular witness for $(\mathscr{A}, \varphi)$.

Let u' (resp. v') denote the data word over Γ obtained from u (resp. v) by replacing each letter $(\gamma, 0)$ (resp. $(\gamma, 1)$) with γ. In the following, we will show that $u'(v')^\omega$ is accepted by $(\mathscr{A}, \varphi)$.

At first, it is easy to see that the run $\rho_u \rho_v^\omega$ is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(u'(v')^\omega)$. For every class X in $u'(v')^\omega$, we will show that

$$\mathsf{Proj}((u'(v')^\omega)|_X) \models \varphi[\mathsf{Parikh}(\mathsf{Proj}((u'(v')^\omega)|_X))].$$

From this, we conclude that $u'(v')^\omega \models_c \varphi$ and $u'(v')^\omega$ is accepted by $(\mathscr{A}, \varphi)$.

Let X be a class in $u'(v')^\omega$ and Y be the class of uv with the same data value as X. Since $uv \models_c \varphi'$, it follows that there is $i : 1 \leq i \leq m$ such that

$$\mathsf{Proj}((uv)|_Y) \models \Big(\bigwedge_{\gamma \in \Gamma} \varphi'_{i,\gamma}\Big)\big[\mathsf{Parikh}(\mathsf{Proj}((uv)|_Y))\big].$$

We will show that $\mathsf{Proj}((u'(v')^\omega)|_X) \models \Big(\bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}\Big)\big[\mathsf{Parikh}(\mathsf{Proj}((u'(v')^\omega)|_X))\big]$.

Let $\gamma \in \Gamma$.

- If $\varphi_{i,\gamma} := x_\gamma = c_{i,\gamma}$, then $\varphi'_{i,\gamma} := x_{(\gamma,0)} = c_{i,\gamma} \wedge x_{(\gamma,1)} = 0$ and $\mathsf{Proj}((uv)|_Y) \models \varphi'_{i,\gamma}$. This implies that the letter $(\gamma, 1)$ does not occur in $v|_Y$, and the number of $(\gamma, 0)$-positions in $u|_Y$ is exactly $c_{i,\gamma}$. Thus the letter γ does not occur in $v'|_Y$ and the number of γ-positions in $u'|_Y$ is exactly $c_{i,\gamma}$. It follows that the number of γ-positions in $(u'(v')^\omega)\,|_X$ is exactly $c_{i,\gamma}$. So $\mathsf{Proj}((u'(v')^\omega)|_X) \models \varphi_{i,\gamma}[\#_{\mathsf{Proj}((u'(v')^\omega)|_X)}(\gamma)]$.

- The discussion for the situation that $\varphi_{i,\gamma} := x_{i,\gamma} \geq p_0 \wedge x_{i,\gamma} \equiv r_{i,\gamma} \bmod p_0$ is similar.
- If $\varphi_{i,\gamma} := x_\gamma = \omega$, then $\varphi'_{i,\gamma} := x_{(\gamma,1)} \geq 1$ and $\mathsf{Proj}((uv)|_Y) \models \varphi'_{i,\gamma}$. This implies that the letter γ occurs at least once in $v'|_Y$, therefore, it occurs infinitely many times in $(u'(v')^\omega)|_X$. It follows that $\mathsf{Proj}((u'(v')^\omega)|_X) \models \varphi_{i,\gamma}[\#_{\mathsf{Proj}((u'(v')^\omega)|_X)}(\gamma)]$.

  Consequently, $\mathsf{Proj}((u'(v')^\omega)|_X) \models \big(\bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}\big)\big[\mathsf{Parikh}(\mathsf{Proj}((u'(v')^\omega)|_X))\big]$.

The situation that uv is a non-singular witness for $(\mathscr{A}, \varphi)$.

Let u' (resp. v') denote the data word over Γ obtained from u (resp. v) by replacing each letter $(\gamma, \dots) \in \Lambda$ with γ.

Consider the data ω-word $u'(v')^\omega$.

At first, we show that all the black and white classes of $u'(v')^\omega$ satisfies φ.

Because $uv \models_c \varphi'$, so $uv \models_c \eta$ and $uv \models_c \psi$. This implies that each class of uv has exactly one color, and for each class X, there is $i : 1 \leq i \leq m$ such that $\mathsf{Proj}((uv)|_X) \models \bigwedge\limits_{\gamma \in \Gamma} \varphi'_{i,\gamma}$.

Let X be a black or white class of $u'(v')^\omega$, Y be the corresponding black or white class of uv, and $i : 1 \leq i \leq m$ such that $\mathsf{Proj}((uv)|_Y) \models \bigwedge\limits_{\gamma \in \Gamma} \varphi'_{i,\gamma}$. We show that $(u'(v')^\omega)|_X$ satisfies $\bigwedge\limits_{\gamma \in \Gamma} \varphi_{i,\gamma}$, thus satisfies φ.

- For any $\gamma \in \Gamma$, if $\varphi_{i,\gamma} := x_{i,\gamma} = c_{i,\gamma}$ or $\varphi_{i,\gamma} := x_{i,\gamma} \geq p_0 \wedge x_{i,\gamma} \equiv r_{i,\gamma} \bmod p_0$, then the class condition $\varphi'_{i,\gamma}$ guarantees that there are no γ-positions in $v'|_Y$ and the number of γ-positions in $u'|_Y$ is either $c_{i,\gamma}$ or no less than p_0 and equal to $r_{i,\gamma}$ modulo p_0. Therefore, $(u'(v')^\omega)|_X$ satisfies $\varphi_{i,\gamma}$.
- For any $\gamma \in \Gamma$, if $\varphi_{i,\gamma} := x_{i,\gamma} = \omega$, then the class condition $\varphi'_{i,\gamma}$ guarantees that there is at least one γ-position in $v'|_Y$. Therefore, there are infinitely many γ-positions in $u'(v')^\omega$. So $(u'(v')^\omega)|_X$ satisfies $\varphi_{i,\gamma}$.

The yellow and blue classes in $u'(v')^\omega$ may not satisfy φ. In the following, we will reassign the data values to the yellow and blue zones of $u'(v')^\omega$ so that all the classes of the resulting data ω-word satisfy φ.

Let $X_{\mathsf{yellow},1}, \dots, X_{\mathsf{yellow},k}$ (where $k \leq |Q|^2$) be a list of all yellow classes in uv (as a matter of fact, all these classes belong to v). Let $\mathsf{Zones}_{\mathsf{fin}}(X_{\mathsf{yellow},i})$ (resp. $\mathsf{Zones}_{\mathsf{inf}}(X_{\mathsf{yellow},i})$) denote the set of "fin"-zones (resp. "inf"-zones) of the class $X_{\mathsf{yellow},i}$ in uv. In addition, for each $i : 1 \leq i \leq k$, let $Y_{\mathsf{yellow},i}$ be the yellow class of $u'v'$ with the same data value as $X_{\mathsf{yellow},i}$, and $\mathsf{Zones}_{\mathsf{fin}}(Y_{\mathsf{yellow},i})$ (resp. $\mathsf{Zones}_{\mathsf{inf}}(Y_{\mathsf{yellow},i})$) be the set of zones in $u'v'$ corresponding to $\mathsf{Zones}_{\mathsf{fin}}(X_{\mathsf{yellow},i})$ (resp. $\mathsf{Zones}_{\mathsf{inf}}(X_{\mathsf{yellow},i})$) in uv.

Suppose $\mathsf{Zones}_{\mathsf{fin}}(Y_{\mathsf{yellow},i}) = \{z^1_{\mathsf{fin},i}, \dots, z^{s_i}_{\mathsf{fin},i}\}$ and $\mathsf{Zones}_{\mathsf{inf}}(Y_{\mathsf{yellow},i}) = \{z^1_{\mathsf{inf},i}, \dots, z^{t_i}_{\mathsf{inf},i}\}$ for each $i : 1 \leq i \leq k$.

For any zone $z^j_{\mathsf{inf},i}$ (where $1 \leq j \leq t_i$), there are infinitely many copies of $z^j_{\mathsf{inf},i}$ in $u'(v')^\omega$. We order these copies from left to right and call them the first, second, ..., copy of $z^j_{\mathsf{inf},i}$ in $u'(v')^\omega$.

Let $d_0, d_1, \dots$ be the data values not occurring in the black or white classes of $u'(v')^\omega$.

Let f be a function from $\mathbb{N}$ to $2^{\mathbb{N}}$ satisfying the following two conditions[1].

- The images of f form a partition of $\mathbb{N}$, namely, $\mathbb{N} = \bigcup\limits_{n \in \mathbb{N}} f(n)$, and for any $n_1, n_2 \in \mathbb{N}$ such that $n_1 \neq n_2$, $f(n_1) \cap f(n_2) = \emptyset$.

[1] Such a function exists. For instance, let $g : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ defined by $g(i,j) = \frac{(i+j)(i+j+1)}{2} + j$, then the function f defined by $f(n) = \{g(n,m) \mid m \in \mathbb{N}\}$ satisfies the two conditions

- For any $n \in \mathbb{N}$, $f(n)$ is an infinite subset of $\mathbb{N}$ that are pairwise non-adjacent, i.e. for any $m_1, m_2 \in f(n)$, it holds that $m_1 \neq m_2 + 1$ and $m_2 \neq m_1 + 1$.

In the following, we first remove all the data values in the yellow and blue classes of $u'(v')^\omega$, so the blue and yellow classes are assumed to have no data values. Then we reassign the data values to the yellow and blue zones from left to right by repeating the following procedure.

In the n-th application, we obtain a new class by assigning the data value d_n to a set of zones $\mathcal{Z}$ such that there is a yellow class X in v satisfying that the zones in $\mathcal{Z}$ correspond to the "fin"-zones of X and infinitely many copies of the "inf"-zones of X which are determined by $f(n)$. More specifically, the procedure goes as follows.
Let z be the first yellow or blue zone (from left to right) in the current data ω-word that does not yet have a data value. We choose a class $Y_{\mathsf{yellow},i}$ of $u'v'$ that contains a zone $z^j_{\mathsf{fin},i}$ (where $1 \le j \le s_i$) such that $\mathsf{a\text{-}Proj}_{u'v'}(z) = \mathsf{a\text{-}Proj}_{u'v'}(z^j_{\mathsf{fin},i})$.
Let $z_1, \dots, z_{j-1}, z_{j+1}, \dots, z_{s_i}$ be a list of zones in in the current data ω-word that do not yet have a data value such that $\mathsf{a\text{-}Proj}_{u'v'}(z_{j'}) = \mathsf{a\text{-}Proj}_{u'v'}(z^{j'}_{\mathsf{fin},i})$ for every $j' : 1 \le j' \le s_i, j' \neq j$. In addition, we require that $z_1, \dots, z_{j-1}, z_{j+1}, \dots, z_{s_i}$ are pairwise non-adjacent. Such a set of zones $z_1, \dots, z_{j-1}, z_{j+1}, \dots, z_{s_i}$ exists since all the zones in $\mathsf{Zones}_{\mathsf{fin}}(Y_{\mathsf{yellow},i})$ of v' repeat infinitely often in $u'(v')^\omega$. Assign the data value d_n to the zones $z_1, \dots, z_{j-1}, z, z_{j+1}, \dots, z_{s_i}$. Moreover, for every zone $z^{j''}_{\mathsf{inf},i} \in \mathsf{Zones}_{\mathsf{inf}}(X_{\mathsf{yellow},i})$ and every $m \in f(n)$, assign the data value d_n to the m-th copy of the zone $z^{j''}_{\mathsf{inf},i}$ in $u'(v')^\omega$.
The new class satisfies φ as the yellow class $X_{\mathsf{yellow},i}$ of uv satisfies φ': Suppose $X_{\mathsf{yellow},i}$ of uv satisfies $\bigwedge\limits_{\gamma \in \Gamma} \varphi'_{i_1,\gamma}$ for some $i_1 : 1 \le i_1 \le m$, then it can be deduced that the new class satisfies $\varphi_{i_1,\gamma}$ for every $\gamma \in \Gamma$. For instance, if $\varphi_{i_1,\gamma} := x_\gamma = \omega$, then

$$\varphi'_{i_1,\gamma} := \left(\bigvee_{\mathsf{color} \in \{\mathsf{black},\mathsf{white}\}} x_{(\gamma,1,\mathsf{color})} \ge 1 \right) \bigvee x_{(\gamma,1,\mathsf{yellow},\mathsf{inf})} \ge 1.$$

From the fact that $X_{\mathsf{yellow},i}$ of uv satisfies $\varphi'_{i_1,\gamma}$, it follows that $X_{\mathsf{yellow},i}$ of uv satisfies $x_{(\gamma,1,\mathsf{yellow},\mathsf{inf})} \ge 1$. This implies that the letter γ occurs in some zone belonging to $\mathsf{Zones}_{\mathsf{inf}}(Y_{\mathsf{yellow},i})$ of $u'v'$. Because the new class contains infinitely many copies for each zone in $\mathsf{Zones}_{\mathsf{inf}}(Y_{\mathsf{yellow},i})$, it follows that the letter γ occurs infinitely many times in the new class, therefore, the new class satisfies that $\varphi_{i_1,\gamma} := x_\gamma = \omega$.

Because the union of the images of f is equal to $\mathbb{N}$ and for each class $Y_{\mathsf{yellow},i}$ of $u'v'$ and each zone $z^j_{\mathsf{fin},i}$ in $\mathsf{Zones}_{\mathsf{fin}}(Y_{\mathsf{yellow},i})$, there are infinitely many copies of $z^j_{\mathsf{fin},i}$ in $u'(v')^\omega$, it follows that for each class $Y_{\mathsf{yellow},i}$ of $u'v'$ and each zone $z^{j'}_{\mathsf{inf},i}$ in $\mathsf{Zones}_{\mathsf{inf}}(Y_{\mathsf{yellow},i})$, all the (infinite) copies of $z^{j'}_{\mathsf{inf},i}$ in $u'(v')^\omega$ will be assigned data values by repeating the above procedure.

Let w denote the resulting data word.

The run $\rho_u \rho_v^\omega$ is an accepting run of $\mathscr{A}$ over $\mathsf{Profile}(w)$. In addition, from the above discussion, we know that $w \models_c \varphi$. Therefore, w is accepted by $(\mathscr{A}, \varphi)$.

The proof of Claim 1 is complete.

D.2 Proof of Claim 2

Suppose there is a data ω-word w accepted by $(\mathscr{A}, \varphi)$. In the following, we will construct a witness uv from w.

Let ρ be an accepting run of $\mathscr{A}$ over the profile of w. The sub-sequence of states that a run ρ takes at the zone borders is called the *zone sub-run* of ρ.

For every zone z in w, adorn z with the a state pair (q, q') such that q, q' are the two states of ρ before and after reading z respectively.

If there are only finitely many classes in w, then it is easy to see that there is a singular witness uv over the alphabet $\Gamma \times \{0, 1\}$ for $(\mathscr{A}, \varphi)$.

So in the following, it is assumed that there are infinitely many classes in w. Then it is easy to see that there are no zones of infinite length in w.

We first introduce some notations.

For any interval $[i, j]$ of a data (finite or ω) word w, let $\mathcal{S}([i, j])$ denote the summary of the data sub-word $w[i, j]$. Let Θ denote the set of all possible summaries, i.e. $\Theta = (\{\geq, <\} \times \{0, \ldots, p_0 - 1\})^{\Gamma}$. For any $\theta \in \Theta$ and $\gamma \in \Gamma$, let $\pi_1(\theta(\gamma))$ (resp. $\pi_2(\theta(\gamma))$) denote the first (resp. second) component of $\theta(\gamma)$. For $\theta_1, \theta_2 \in \Theta$, we can define $\theta_1 + \theta_2$ as follows: For each $\gamma \in \Gamma$, if $\pi_1(\theta_1(\gamma)) = \geq$ or $\pi_1(\theta_2(\gamma)) = \geq$ or $\pi_2(\theta_1(\gamma)) + \pi_2(\theta_2(\gamma)) \geq p_0$, then $(\theta_1 + \theta_2)(\gamma) = (\geq, (\pi_2(\theta_1(\gamma)) + \pi_2(\theta_2(\gamma))) \mod p_0)$, otherwise, $(\theta_1 + \theta_2)(\gamma) = (<, \pi_2(\theta_1(\gamma)) + \pi_2(\theta_2(\gamma)))$.

Let X be a class of a data ω-word w. Define $\Gamma_F(X)$ as the set of letters $\gamma \in \Gamma$ that occur only finitely many times in X. Similarly, define $\Gamma_I(X)$ as the set of letters $\gamma \in \Gamma$ that occur infinitely many times in X. Define $\mathsf{Zones}_F(X)$ as the set of zones z of X such that z contains at least one γ-position with $\gamma \in \Gamma_F(X)$, and define $\mathsf{Zones}_I(X)$ as the set of all the other zones of X. Note that there are only finitely many zones in $\mathsf{Zones}_F(X)$, and if $z \in \mathsf{Zones}_I(X)$, then every γ-position in z satisfy that $\gamma \in \Gamma_I(X)$. We call the zones in $\mathsf{Zones}_F(X)$ as the 'fin"-zones of X, and zones in $\mathsf{Zones}_I(X)$ as the 'inf"-zones of X. In addition, define the *finitary summary* of X, denoted by $\mathcal{S}_F(w|_X)$, as the function $\theta : \Gamma_F(X) \to \{\geq, <\} \times \{0, \ldots, p_0 - 1\}$ defined as follows,

- if $\#_{\mathsf{Proj}(w|_X)}(\gamma) < p_0$, then $\theta(\gamma) = (<, \#_{\mathsf{Proj}(w|_X)}(\gamma))$,
- if $\#_{\mathsf{Proj}(w|_X)}(\gamma) \geq p_0$, then $\theta(\gamma) = (\geq, \#_{\mathsf{Proj}(w|_X)}(\gamma) \mod p_0)$.

The construction is divided into three transformation steps.

Step 1.

At first, we transform w to a data ω-word in which every zone is of length at most $|Q|(2p_0)^{|\Gamma|}$.
If a zone z with adorned state pair (q, q') is of length $l \geq |Q|(2p_0)^{|\Gamma|} + 1$, let $q_1 \ldots q_{l+1}$ be the state sequence of ρ restricted to z such that $q_1 = q$ and $q_{l+1} = q'$. In addition, for each $i : 1 \leq i \leq l + 1$, let θ_i denote the summary of the interval $[1, i - 1]$ of z (if $i = 1$, then $z[1, i - 1] = \varepsilon$). Because $l \geq |Q|(2p_0)^{|\Gamma|} + 1$, it follows that there are $i, j : 2 \leq i < j \leq l + 1$ such that $(q_i, \theta_i) = (q_j, \theta_j)$. Therefore, the summary θ of the interval $[i, j - 1]$ in z satisfies that for every $\gamma \in \Gamma$, $\theta(\gamma) = (s, 0)$ for some $s \in \{\geq, <\}$. Then the data subword $z[i, j - 1]$ can be removed from z without affecting the zone sub-run of ρ and the summary of z. This procedure can be repeated, until we get a zone of length at most $|Q|(2p_0)^{|\Gamma|}$.
Let w_1 denote the resulting data ω-word.
For each class X of w_1, select a set of at most $(2p_0)^{|\Gamma|}$ zones in $\mathsf{Zones}_F(X)$, denoted by $\mathcal{Z}$, such that $\mathcal{S}_F(w_1|_X) = \sum\limits_{z \in \mathcal{Z}} \left(\mathcal{S}(z)|_{\Gamma_F(X)}\right)$, where $\mathcal{S}(z)|_{\Gamma_F(X)}$ is the restriction of the summary $\mathcal{S}(z)$ to $\Gamma_F(X)$. Call these zones as the core *"fin"-zones of X, and all the other zones in $\mathsf{Zones}_F(X)$ as the* redundant *"fin"-zones of X.*

For each class X of w_1, the set of zones $\mathcal{Z}$, satisfying the above condition, exists: If there are at most $(2p_0)^{|\Gamma|}$ zones in $\mathsf{Zones}_F(X)$, then let $\mathcal{Z} := \mathsf{Zones}_F(X)$. Otherwise, there are at least $\big((2p_0)^{|\Gamma|}+1\big)$ zones in $\mathsf{Zones}_F(X)$. Consider any $\big((2p_0)^{|\Gamma|}+1\big)$ pairwise-distinct zones $z_1,\dots,z_{(2p_0)^{|\Gamma|}+1} \in \mathsf{Zones}_F(X)$. By the pigeon hole principle, there must be $1 \leq i < j \leq (2p_0)^{|\Gamma|}+1$ such that $\mathcal{S}(z_1)+\cdots+\mathcal{S}(z_i) = \mathcal{S}(z_1)+\cdots+\mathcal{S}(z_j)$. Let $\theta = \mathcal{S}(z_{i+1})+\cdots+\mathcal{S}(z_j)$, then $\pi_2(\theta(\gamma)) = 0$ for every $\gamma \in \Gamma$. It follows that $\mathcal{Z} = \mathsf{Zones}_F(X) \setminus \{z_{i+1},\dots,z_j\}$ is a proper subset of $\mathsf{Zones}_F(X)$ satisfying that $\mathcal{S}_F(w_1|_X) = \sum\limits_{z \in \mathcal{Z}} (\mathcal{S}(z)|_{\Gamma_F(X)})$. Continue like this, we finally get a desired $\mathcal{Z}$ containing at most $\big((2p_0)^{|\Gamma|}\big)$ zones.

In the following, we first give an overview of the two remaining transformation steps, then present them in detail.

For each tuple (θ, q, q') such that there is a "fin" (resp. "inf") zone in w_1 with the summary θ and the adornment (q,q'), select a bounded number of classes in w_1 which contain such a zone. These classes are assigned the black color. For the non-black classes, we distinguish between the state pairs (q,q') that occur as the adornment of only a finite number of non-black zones and those that occur as the adornment of infinitely many non-black zones. The former are called as infrequent adornments and the latter as frequent adornments. Those classes which contain a zone with some infrequent adornment (q,q') are assigned the white color. Then a position p_1 is selected such that all the "fin"-zones in the black and white classes are before p_1. For each state pair (q,q') such that there is a non-black and non-white "fin"-zone with the (frequent) adornment (q,q'), select one class located after p_1 that contains such a zone. These classes are assigned the yellow color. All the remaining classes are assigned the blue color. In addition, each white or yellow or blue class is trimmed by applying some replacements to the "inf"-zones in the class. Finally a position p_2 after p_1 is selected and a non-singular witness is obtained.

Now we present the remaining transformation steps in detail.

Step 2.

For every summary $\theta \in (\{\geq,<\} \times \{0,\dots,p_0\})^{|\Gamma|}$ and state pair (q,q'),

- *if there are redundant "fin"-zones in w_1 with the summary θ and the adornment (q,q'), choose (up to) $\big(2(2p_0)^{|\Gamma|}+1\big)$ classes containing such redundant "fin"-zones;*
- *if there are "inf"-zones in w_1 with the summary θ and the adornment (q,q'), choose (up to) three classes containing such "inf"-zones.*

Assign the black color to these classes.
For each non-black class X, if there are at least $\big(|Q|^2(2p_0)^{2|\Gamma|}+1\big)$ redundant "fin"-zones in X, then we choose $k \leq (2p_0)^{|\Gamma|}$ redundant "fin"-zones in X, say $z_1,\dots,z_k$, such that all z_i's have the same summary (say θ) and the same adornment (say (q,q')), and for each $\gamma \in \Gamma$, $k\pi_2(\theta(\gamma)) \equiv 0 \bmod p_0$. Let D be the set of data values occurring in the zones that are adjacent to at least one of $z_1,\dots,z_k$. Select a black "fin"-zone z with the summary θ and the adornment (q,q') such that the data value occurring in z is different from the data values in D. Replace every zone z_i $(1 \leq i \leq k)$ by z. Note that these replacements do not change the zone sub-run as well as the finitary summaries of classes.
After these replacements, for every non-black class X, $\mathsf{Zones}_F(X)$ contains only at most $(2p_0)^{|\Gamma|}$ core "fin"-zones and at most $|Q|^2(2p_0)^{2|\Gamma|}$ redundant "fin"-zones.

Let w_2 be the resulting data ω-word.

For each class X containing at least $\big(|Q|^2(2p_0)^{2|\Gamma|}+1\big)$ redundant "fin"-zones, the zones $z_1,\dots,z_k$ satisfying the above conditions exist. The argument goes as follows: From the pigeon hole principle, there are $\theta \in \Theta$ and $(q, q') \in Q^2$ such that at least $\big((2p_0)^{|\Gamma|} + 1\big)$ redundant "fin"-zones in X have the summary θ and the adornment (q, q'). Let $z_1, \ldots, z_n$ (where $n \geq (2p_0)^{|\Gamma|} + 1$) be a list of all these redundant "fin"-zones with the summary θ and the adornment (q, q'). Then by the pigeon hole principle again, there are $k \leq (2p_0)^{|\Gamma|}$ such that for each $\gamma \in \Gamma$, $k\pi_2(\theta(\gamma)) \equiv 0 \bmod p_0$. Then $z_1, \ldots, z_k$ are the desired zones.

In addition, for zones $z_1, \ldots, z_k$ with the summary θ and adornment (q, q'), because $|D| \leq 2(2p_0)^{|\Gamma|}$ and there are at least $(2(2p_0)^{|\Gamma|} + 1)$ black "fin"-zones with the summary θ and the adornment (q, q'), it follows that a desired black "fin"-zone z with a data value different from those in D can be selected to replace all of $z_1, \ldots, z_k$.

For any state pair (q, q'), if there are infinitely many non-black (core or redundant) "fin"-zones with the adornment (q, q'), then call (q, q') a *frequent adornment*, otherwise an *infrequent adornment*. There are only finitely many non-black classes containing "fin"-zones with infrequent adornments. Assign the white color to these classes.

Since it is assumed that there are infinitely many classes in w, there must be (infinitely many) classes which are neither black nor white. For any non-black and non-white "fin"-zone, its adornment (q, q') must be frequent.

Similar to the proof for weak data automata in [11], without loss of generality, we assume that

if in the run ρ, an accepting state is reached when reading a position in the middle of a zone z (of finite length), then an accepting state is also reached after reading the last position of z.

Let $\hat{q}$ be a state in F occurring infinitely often in the zone sub-run. Let p_1 be the minimal position satisfying the following conditions,

- p_1 is the last position of a zone,
- the zone sub-run assumes $\hat{q}$ in the beginning of the position $p_1 + 1$,
- and for each black or white class X, all the zones in $\mathsf{Zones}_F(X)$ occur before p_1.

Step 3.

For each frequent adornment (q, q'), select a non-black and non-white class $X_{q,q'}$ such that $X_{q,q'}$ contains a "fin*"-zone with the adornment (q, q') and all the positions in $X_{q,q'}$ are after p_1. Such a class exists since there are infinitely many non-black and non-white "*fin*"-zones with the adornment (q, q'). Assign the yellow color to these classes $X_{q,q'}$.*
Assign the blue color to all the classes that have not been assigned a color.
For each white or yellow class X, select at most $|\Gamma|$-zones in $\mathsf{Zones}_I(X)$ such that all these zones are after p_1 and each letter in $\Gamma_I(X)$ occurs at least once in these zones. Call these zones as the core *"*inf*"-zones of X and all the other zones in $\mathsf{Zones}_I(X)$ as the* redundant *"*inf*"-zones of X.*
For every white or yellow class X and every redundant "inf*"-zone z in X, let z' be a black "*inf*"-zone such that z' has the same summary and the same adornment as z and has a data value different from the data values of the two zones adjacent to z. Replace z by z'.*
After these replacements, there are no redundant "inf*"-zones in each white or yellow class.*
For each blue class X and each "inf*"-zone z in X, let z' be a black "*inf*"-zone such that z' has the same summary and the same adornment as z and has a data value different from the data values of the two zones adjacent to z. Replace z by z'.*

*After these replacements, there are no "*inf*"-zones in each blue class.*
*For each black class X, select at most $|\Gamma|$ "*inf*"-zones such that all these zones are after p_1 and each letter in $\Gamma_I(X)$ occurs at least once in these zones. Call these zones as the* core *"*inf*"-zones of X and all the other "*inf*"-zones as the* redundant *"*inf*"-zones of X.*
*For each blue "*fin*"-zone z with adornment (q, q'), let z' be a yellow "*fin*"-zone with the same adornment as z. Keep the data value of z and replace the string projection of z by that of z'.*

Let w_3 be the resulting data ω-word.
Let p_2 be the minimal position after p_1 in w_3 such that

- p_2 is the last position of a non-black and non-white zone in w_3,
- the zone sub-run assumes the state $\hat{q}$ in the beginning of the position $p_2 + 1$,
- for each black or white class X, all the core "inf"-zones of X occur before p_2,
- for each yellow class X, all the "fin"-zones of X and all the core "inf"-zones of X occur before p_2.

Such a position p_2 exists since there are infinitely many non-black and non-white classes in w_3 and $\hat{q}$ occurs infinitely often in the zone-sub run.
Let u' be the prefix of w_3 until the position p_1 and v' be the sub-string of w_3 from the position $p_1 + 1$ to the position p_2.
Let u be the data word obtained from u' by replacing every letter (γ, color) with $(\gamma, 0, \mathsf{color})$ and v be the data word obtained from v' by applying the following replacements,

- for every $\mathsf{color} \in \{\mathsf{black}, \mathsf{white}, \mathsf{blue}\}$, replace every letter (γ, color) with $(\gamma, 1, \mathsf{color})$,
- replace every letter $(\gamma, \mathsf{yellow})$ occurring in the "fin"-zones of w_3 by $(\gamma, 1, \mathsf{yellow}, \mathsf{fin})$ and every letter $(\gamma, \mathsf{yellow})$ occurring in the "inf"-zones of w_3 by $(\gamma, 1, \mathsf{yellow}, \mathsf{inf})$.

From the construction, it is not hard to see that $uv \models_c \varphi'$. Thus, uv is a non-singular witness for $(\mathscr{A}, \varphi)$.
The proof of Claim 2 is complete.

D.3 Proof of Claim 3

We only illustrate the proof by presenting a nondeterministic algorithm to construct from $(\mathscr{A}, \varphi)$ a CDA $(\mathscr{B}, \psi)$ which accepts non-singular witnesses. The construction of a CDA to accept singular witnesses is simpler and similar.

At first, the nondeterministic algorithm guesses an accepting state $\hat{q}$, a number $\ell \leq |Q|^2$, and a sequence of adorned projections of the "fin"-zones in the ℓ yellow classes (from left to right), which is of the length

$$\ell \left((2p_0)^{|\Gamma|} + |Q|^2 (2p_0)^{2|\Gamma|} \right) |Q| (2p_0)^{|\Gamma|}.$$

The size of the sequence follows from the fact that there are ℓ yellow classes, each yellow class contains at most $\left((2p_0)^{|\Gamma|}\right)$ core "fin"-zones and at most $\left(|Q|^2(2p_0)^{2|\Gamma|}\right)$ redundant "fin"-zones, and each yellow "fin"-zone is of the size at most $|Q|(2p_0)^{|\Gamma|}$.

In addition, the algorithm guesses for each yellow "fin"-zone, an index $i : 1 \leq i \leq \ell$, with the intention that the set of yellow "fin"-zones with the index i is the set of "fin"-zones in the i-th yellow class.

Then it constructs $\mathscr{B}$ over the alphabet

$$\Gamma \times \{0,1\} \times \{\mathsf{black}, \mathsf{white}, \mathsf{blue}\} \bigcup$$
$$\Gamma \times \{1\} \times \{\mathsf{yellow}\} \times \{\mathsf{fin}\} \times \{1, 2, \ldots, \ell\} \bigcup \Gamma \times \{1\} \times \{\mathsf{yellow}\} \times \{\mathsf{inf}\}$$

as follows.

- Roughly speaking, $\mathscr{B}$ checks all the conditions in the definition of non-singular witnesses except the condition that $uv \models_c \varphi'$ and the condition that the set of yellow "fin"-zones with the same index are indeed in the same class.
  - At first, $\mathscr{B}$ simulates $\mathscr{A}$ over $\mathsf{Profile}(u'v')$ and verifies that the two states of $\mathscr{A}$ after reading $\mathsf{Profile}(u')$ and after reading $\mathsf{Profile}(u'v')$ are both $\hat{q}$, where $u'v'$ is obtained from uv by replacing each letter $(\gamma, \dots) \in \Lambda$ with γ.
  - It remembers the guessed adorned projections of the yellow zones, and checks whether the adorned projections of the yellow zones in uv are those that have been guessed; in addition, it checks whether for each blue zone, there is a yellow zone with the same adorned projection.
  - $\cdots$
- ψ is a conjunction of the following two formulas,
  - the formula φ'' obtained from φ' (in the definition of non-singular witnesses) by replacing each variable $x_{(\gamma,b,\mathsf{yellow},\mathsf{fin})}$ with $\sum\limits_{1 \le i \le \ell} x_{(\gamma,b,\mathsf{yellow},\mathsf{fin},i)}$,
  - the formula η which asserts that the guessed yellow "fin"-zones with the same index are indeed in the same class: Let $\Lambda'_{\mathsf{yellow},i} = \Gamma \times \{1\} \times \{\mathsf{yellow}\} \times \{\mathsf{fin}\} \times \{i\}$ for each $i : 1 \le i \le \ell$, and $\Lambda'_{\mathsf{yellow}} = \bigcup\limits_{1 \le i \le \ell} \Lambda'_{\mathsf{yellow},i}$, then

$$\eta := \bigwedge_{1 \le i \le \ell} \left(\left(\bigvee_{\lambda' \in \Lambda'_{\mathsf{yellow},i}} x_{\lambda'} \ge 1 \right) \to \bigwedge_{j \ne i} \left(\bigwedge_{\lambda' \in \Lambda'_{\mathsf{yellow},j}} x_{\lambda'} = 0 \right) \right).$$

D.4 Complexity analysis and the general case that φ is not in the normal form

From the proof of Claim 3, the nondeterministic algorithm first guesses a sequence of adorned projections of the yellow zones of the length $O(|Q|^2 \left((2p_0)^{|\Gamma|} + |Q|^2(2p_0)^{2|\Gamma|}\right) |Q|(2p_0)^{|\Gamma|})$, then constructs a CDA $(\mathscr{B}, \psi)$ that accepts the witnesses for $(\mathscr{A}, \varphi)$.

Since $\mathscr{B}$ need remember the guessed sequences, the number of states of $\mathscr{B}$ is

$$O\left(|Q|^2 \left((2p_0)^{|\Gamma|} + |Q|^2(2p_0)^{2|\Gamma|}\right) |Q|(2p_0)^{|\Gamma|}\right).$$

In addition, the size of ψ is $O\left(|\varphi''| + 2|\Gamma||Q|^4\right) = O\left(|Q|^2|\varphi'| \log |Q| + 2|\Gamma||Q|^4\right)$.

Because $p_0 \le 2^{|\varphi|}$ and the size of φ' is $O(|\varphi|)$, it follows that the size of $\mathscr{B}$ is $O(|Q|^5)2^{O(|\varphi||\Gamma|)}$ and the size of ψ is $O(|Q|^2(|\varphi| \log |Q| + 2|\Gamma||Q|^2))$.

Therefore, the nondeterministic algorithm to construct a CDA $(\mathscr{B}, \psi)$ that accepts the witnesses for $(\mathscr{A}, \varphi)$ runs in exponential time.

Finally, we consider the general case that φ is not in the normal form.

From Proposition 13, we know that a normal form of φ of size $2^{O(|\Gamma||\varphi|)}$ can be obtained from φ in exponential time; in addition, p_0, the normalization number of φ, is at most $2^{|\varphi|}$.

Then from the above analysis, we know that if the formula φ is not in the normal form, then the nondeterministic algorithm to construct a CDA $(\mathscr{B}, \psi)$ still runs in exponential time.

Thus, there is a NEXPTIME-reduction from the nonemptiness problem of CBDA to the nonemptiness problem of CDA. On the other hand, from Theorem 7, the nonemptiness of CDA can be solved in 3-NEXPTIME.

We conclude that the nonemptiness of CBDA can be solved in 4-NEXPTIME.